Docker

Why should I care ?

Muhammad Falak R Wani

I assume no prior knowledge*.

Agenda

- Part 0: **Foundations**
- Part I: **Basics**
- Part II: **Intermediate**

Starting with first principle

Foundations

Building the Ground

- Linux Architecture
- Role Of the Kernel
- Role Of the User-Space
- Virtualization Techniques
- Using Chroot
- Using Qemu
- Analogy of virtualenv (Python)

Basics

- Using Docker
- Container
- Containers are Stateless ? (bug/feature)
- Images
- Docker Files
- Some Use-cases
- DockerHub (analogy github.com)

Intermediate

- Internals of docker run
- Making GUI work in docker
- Making Sound work in docker
- Docker Compose

Let's cover 33.33333333 things

You do the math.

Foundations

Linux Architecture I

```
ls /
```

```
bin boot data dev etc home
lib proc root sys tmp usr

.. some uninteresting dirs skipped
```

What Happens when you type a command @ the $SHELL

- A search in the $PATH variable
- Stop on the first match found
- The Type of executeable determined (**file**)
- Some *insane* mechanism executes it
- Return To the $SHELL prompt

Linux Architecture II

The system (*nix) has two parts

- The user-space (programs, including the $SHELL)
- The kernel (Actual OS)

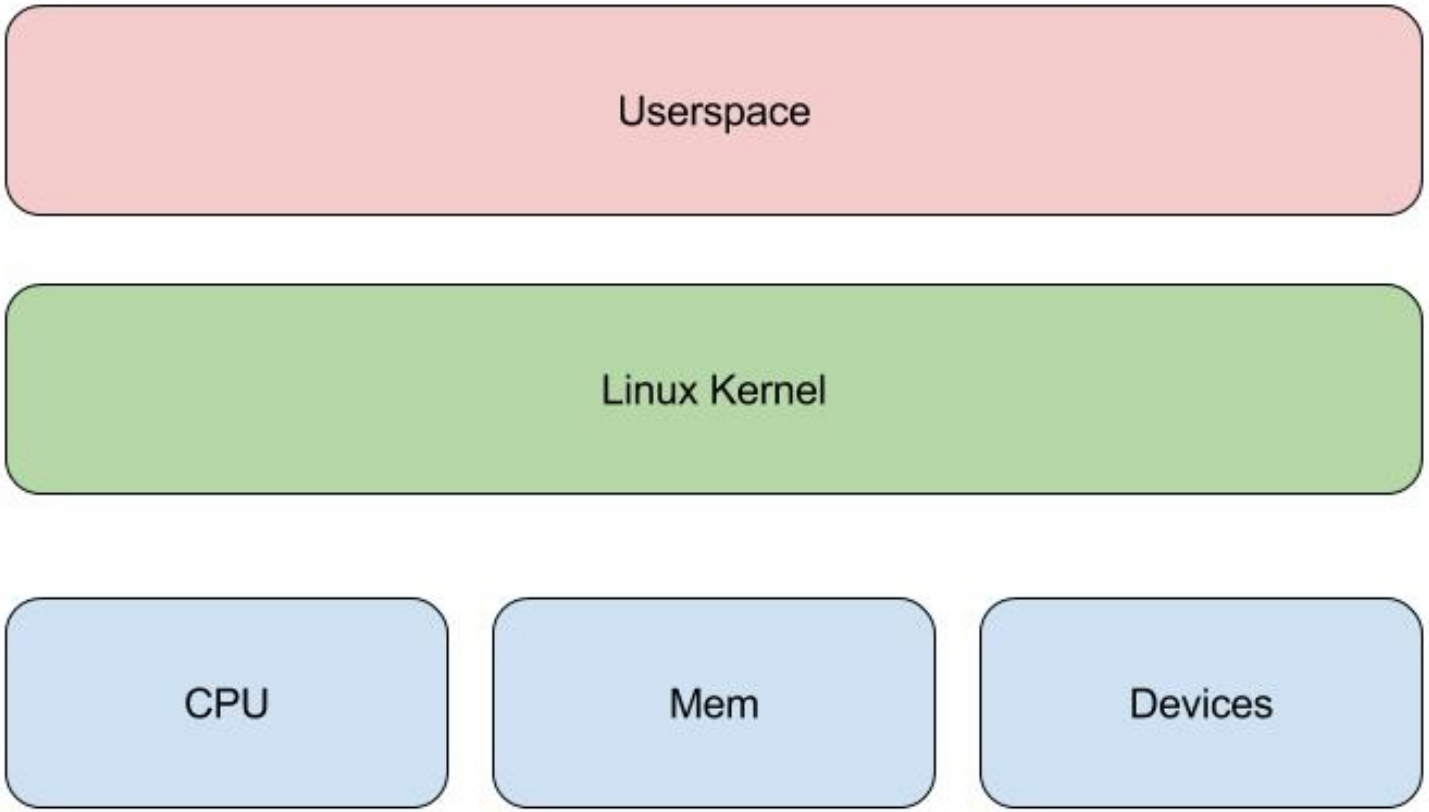
Both are essential and cannot exist* without each-other
BUT
Both are **Plugable/Swapable** !

```
Hardware is stateless !!
```

What makes your Computer Statefull ?

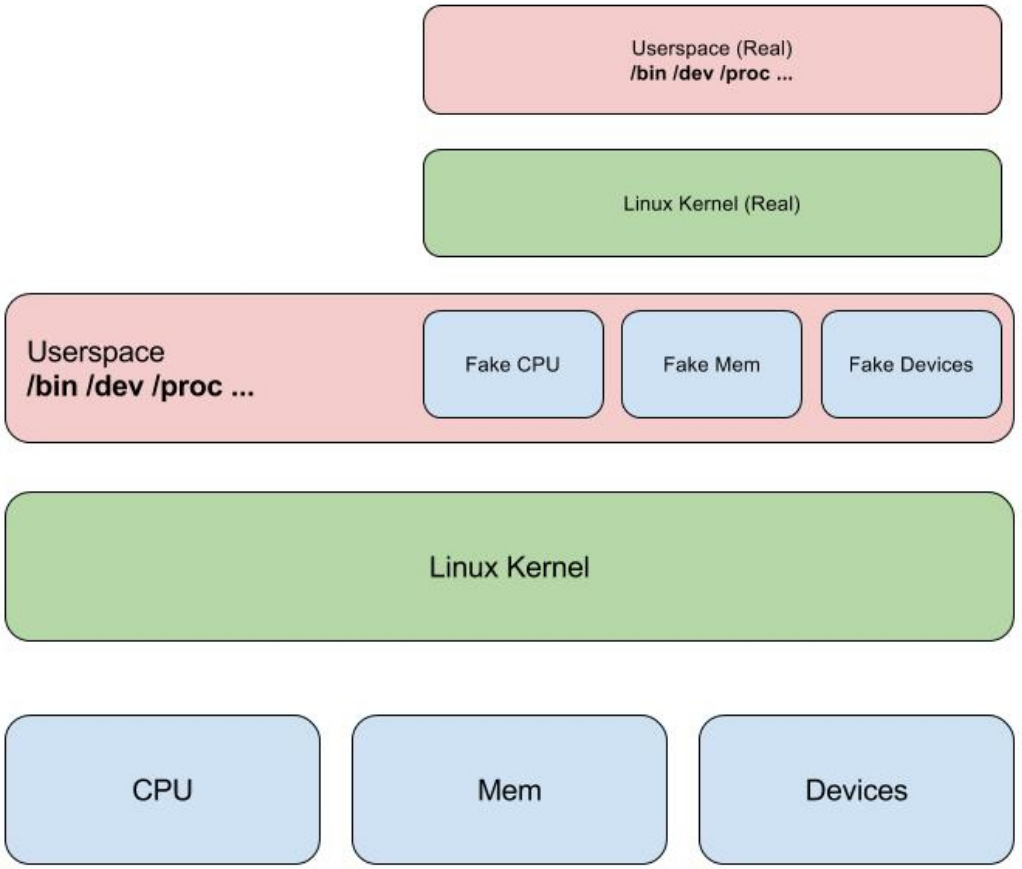
Linux Architecture III

Normal System



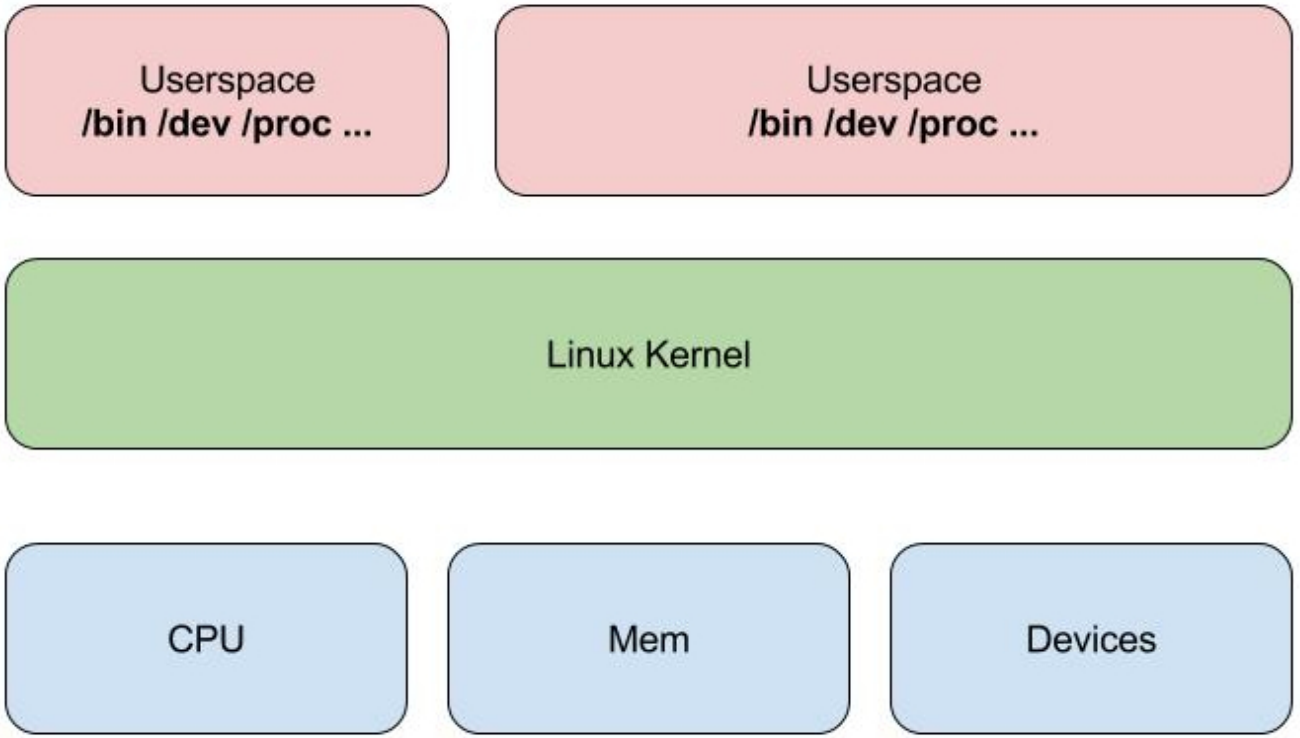
Linux Architecture IV

Hypervisor Type II (Virtual Box)



Linux Architecture V

Docker



Chroot

Unix v7 in 1979: **Bill Joy**

- Download **busybox**
- Compile it ...

```
#!/bin/sh

sudo chroot busy-box /bin/sh
# busy-box => path to busy box install
# /bin/sh  => Command to execute in new root (will run relative to new root)
```

- Securit Risk :(
- No Real Isolation :(

Sandboxes


Alison Macrina
@flexlibris
Following
I need to kno how developers came up w the idea of "sandbox" as "a thing that isolates stuff really well"
I mean
have you seen a sandbox
10:15 AM - 30 Jun 2016
1,172 Retweets 1,872 Likes
97
1.2K
1.9K

Using qemu

- Quick Emulator

```
qemu-system-x86_64 -boot d -cdrom random_os.iso -m 2048M -smp 4 -enable-kvm
```

- -enable-kvm => Uses features of new CPUs to speed up.
- -smp => The number of cores
- -boot => boot from hda/dvd etc etc
- -m => RAM

Qemu cpu emulator

qemu-system emulates a full system

qemu-arm (qemu-x86, qemu-ppc ...) emulate only the cpu, so you can run commands

```
qemu-arm some_random_arm_binary
```

Unrelated Fact (30-Aug "17)

Yesterday Acutally .. As if you knew ;)

One last analogy

virtualenv (python)

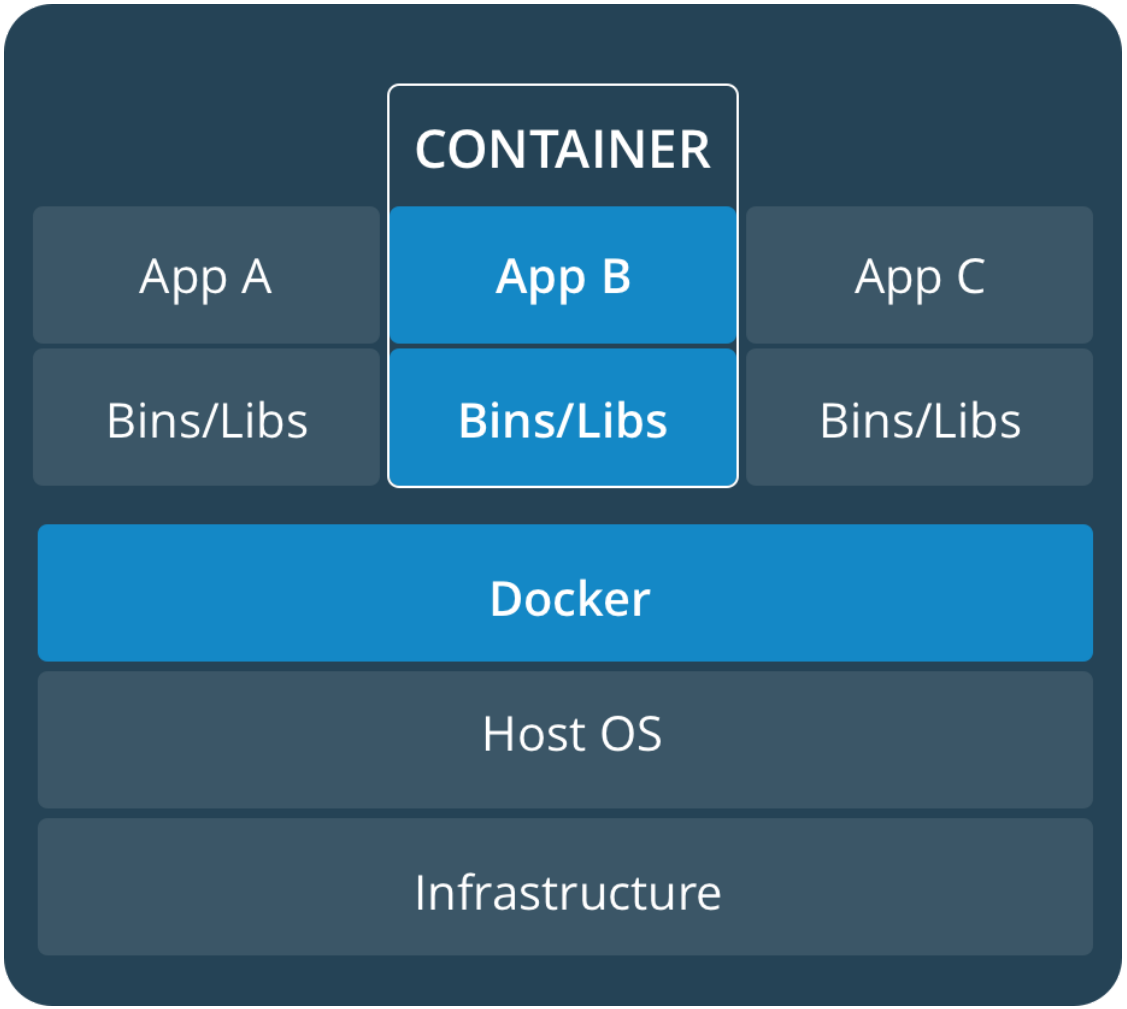
It changes the shells interpretation of python

- You can always work with weired setups
- You dont distrub your system
- Python2/Python3 .. wont give you any pain

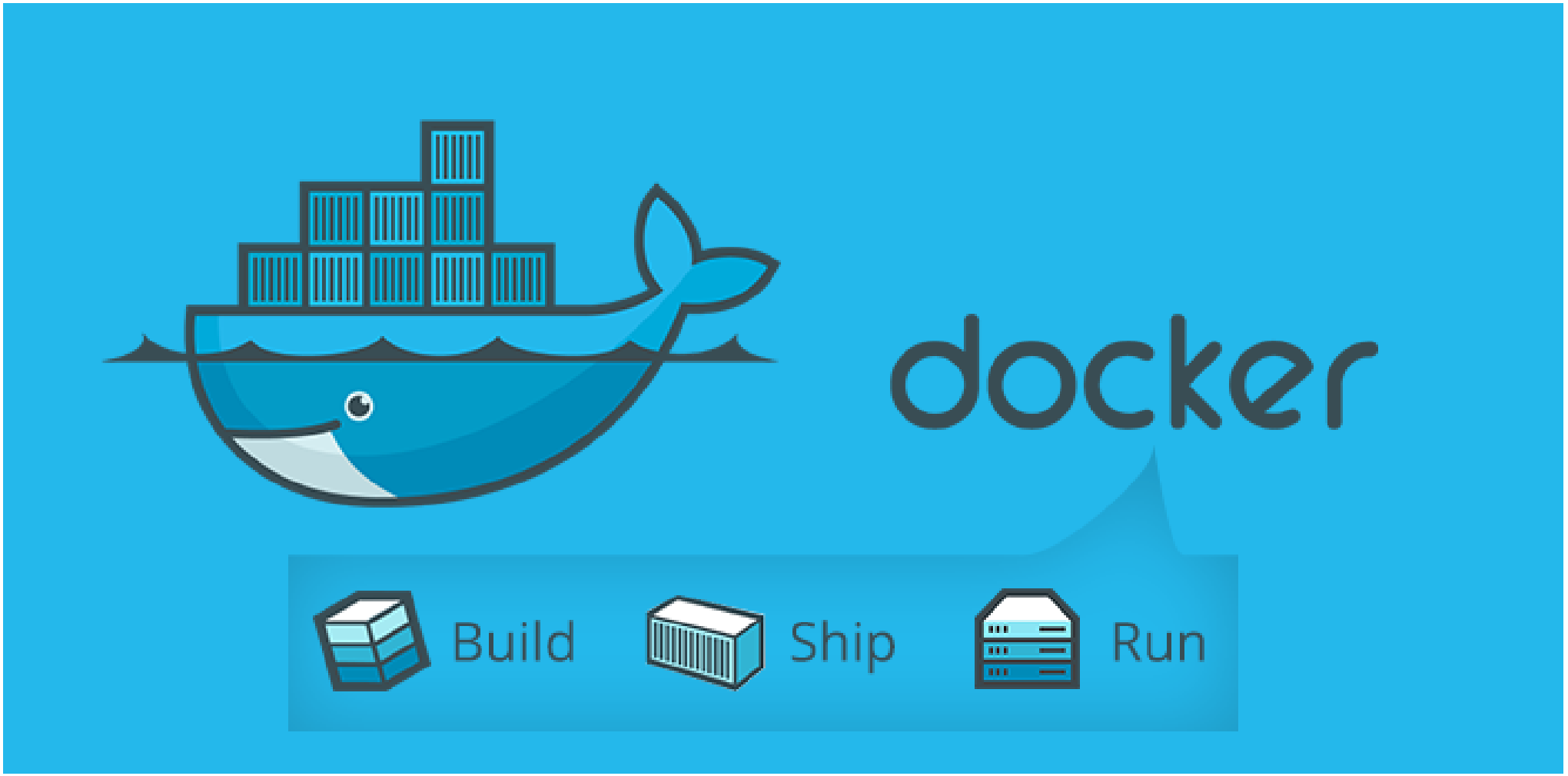
```
virtualenv -p python3 py3
#creates a folder py3
source py3/bin/activate
#note your prompt changes on bash
#use pip install without root

deactivate
#get out from the venv
```

Buzz Word: Container

How does it look like



Forget Configurations



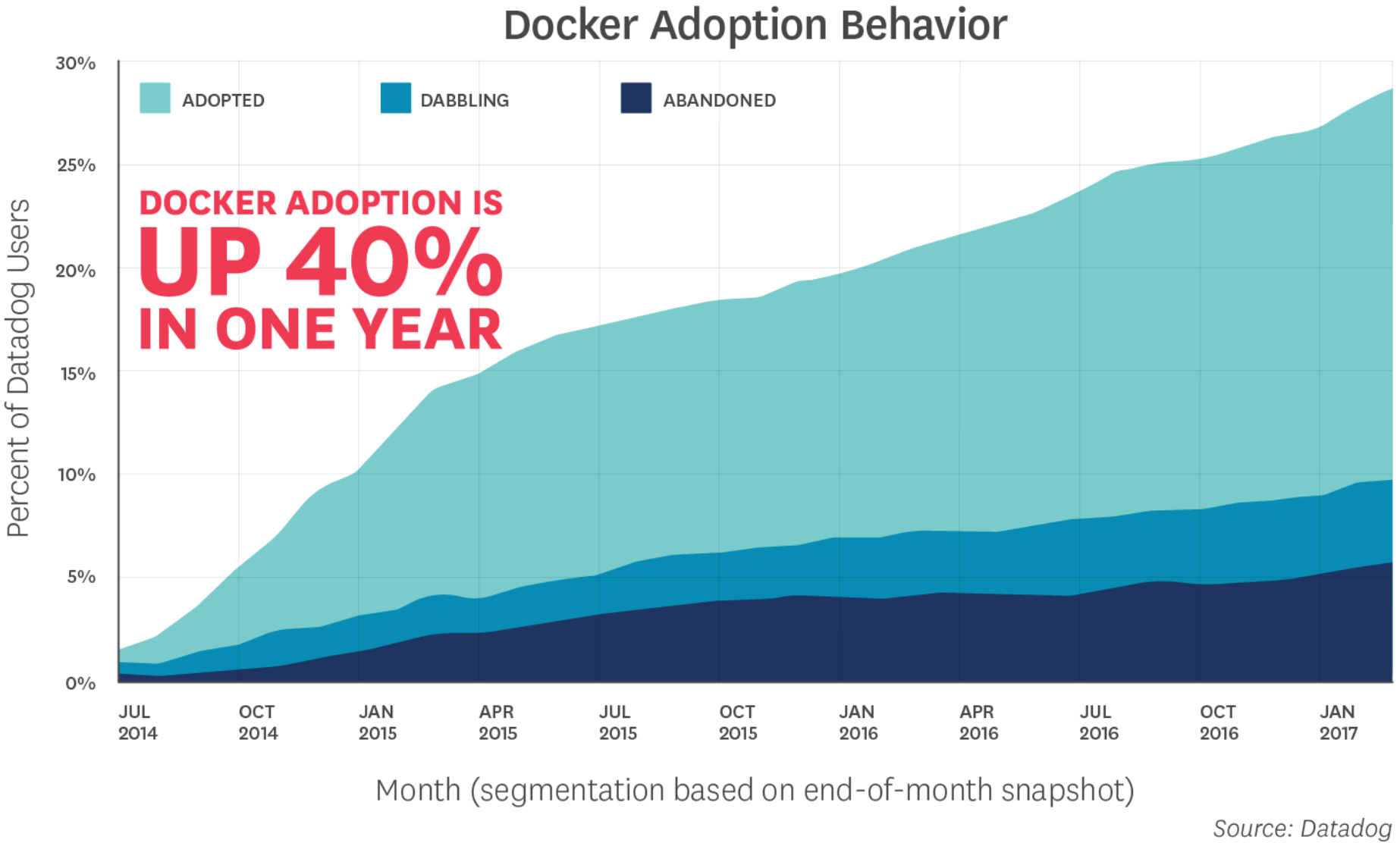
What is a container

- An isolated environment (similar as a vm)
- In Linux it is implemented by using cgroups and namespaces
- NO SUCH THING as a CONTAINER in the kernel
- Its just a namesake given to an isolated environment.
- You dont just ship your code, but package the depencies and configuration.
- No more dependcy hell.
- Stuff just works magically.

Solves the problem of:

```
Works on my machine
```

Why exactly should I care?



Docker

Install:

https://docs.docker.com/engine/installation (https://docs.docker.com/engine/installation/)

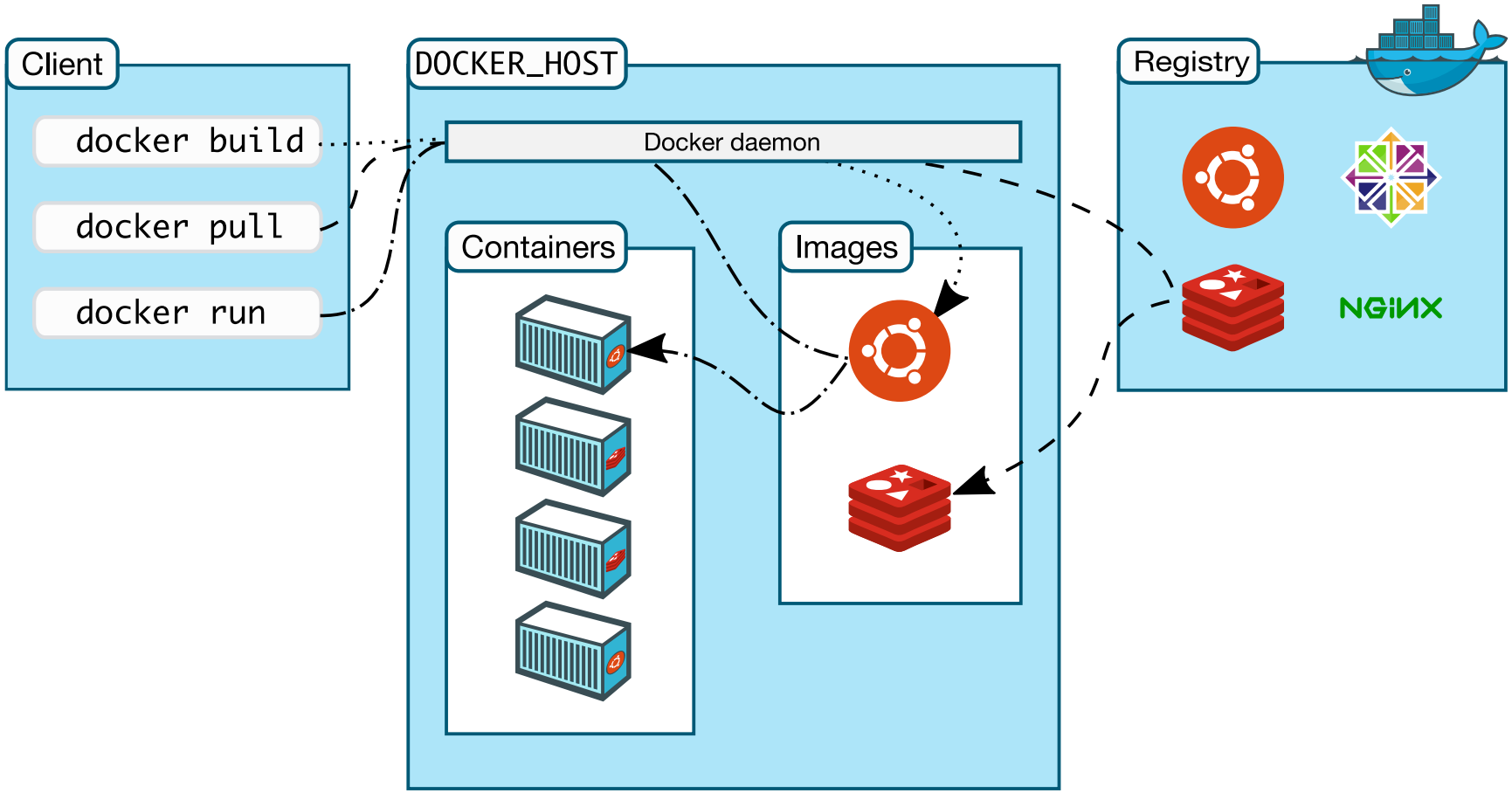
```
sudo usermod -aG username docker
```

First steps:

```
docker container run hello-world
```

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
b04784fba78d: Pulling fs layer
b04784fba78d: Download complete
b04784fba78d: Pull complete
Digest: sha256:f3b3b28a45160805bb16542c9531888519430e9e6d6ffc09d72261b0d26ff74f
Status: Downloaded newer image for hello-world:latest

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

What Just happened



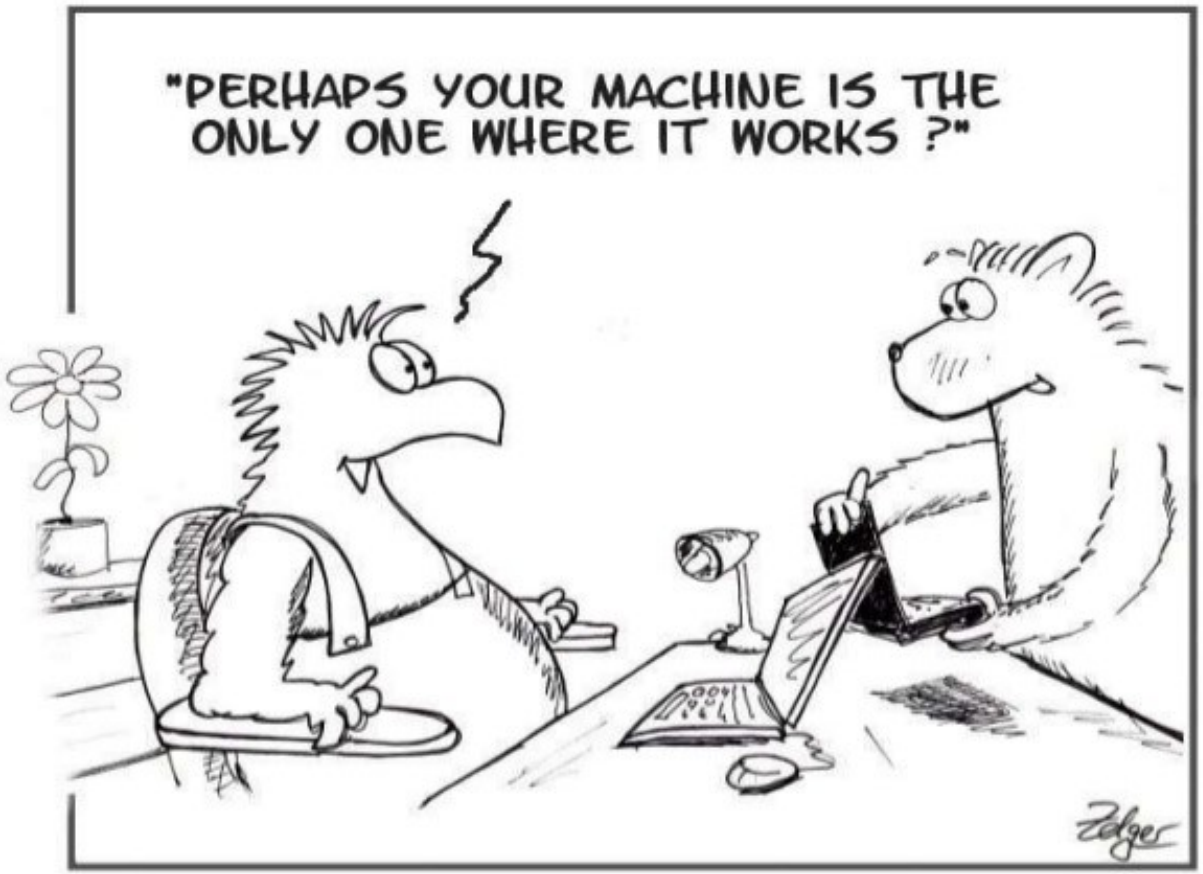
What Just happened

- You did not have the hello-world image so docker downloaded it from a **registry**
- Using the hello world image, a container was provisioned
- The code inside the container executed.
- The code finished, the container exited.

```
Its still lying around unless you explicitly remove it
```

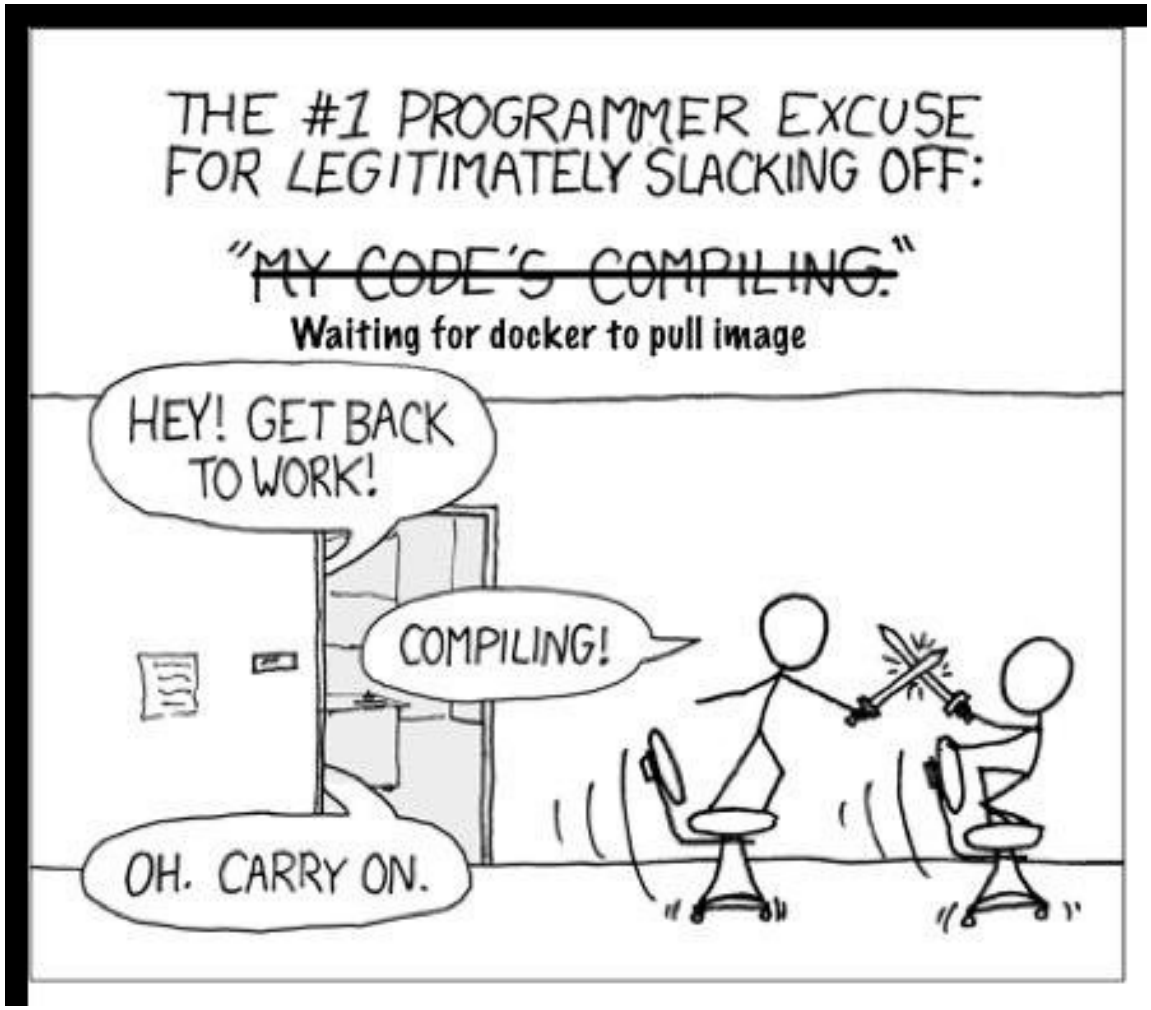
- You can restart a stopped container.

When you ship code!



It works on my machine

Slack time


THE #1 PROGRAMMER EXCUSE FOR LEGITIMATELY SLACKING OFF:
"MY CODE'S COMPILING."
Waiting for docker to pull image
HEY! GET BACK TO WORK!
COMPILING!
OH. CARRY ON.

Docker essentials

- docker **ps** (show what containers are running)
- docker **images** (show your local images)
- docker **run** (run an image)
- docker **pull** (get an image from a registry)
- docker **rm** (remove a stopped container)
- docker **rmi** (remove an image)
- docker **exec** (run a command in a running container)

!so essentials:

- docker info
- docker inspect
- docker network
- .....

docker run

Interactive usage (CTRL-P + CTRL-Q to detach)

```
docker run -it image-name [command]
-i => Keep STDIN open even if not attached
-t => Allocate a pseudo-tty
```

Detached usage

```
docker run -d image-name [command]
-d => Run container in background and print container ID
```

Other flags

```
--cpu-shares   => CPU shares (relative weight)
--hostname     => Container host name
--memory       => Memory limit
-p, --publish  => Publish a container's port(s) to the host
-v, --volume   => Bind mount a volume
-w, --workdir  => Working directory inside the container
```

docker run --help | docker container run --help

docker ps

Show running containers:

```
docker ps
```

Show all containers running and Exited:

```
docker ps -a
```

Restart a stopped container

```
docker start #hash
```

Clean all stopped containers:

```
docker rm $(docker ps -aq)
```

Things to remember

- A container in itself is a full machine (s/w) without kernel
- A container has its own IP address.
- Its own networking interface.
- Is completely* Isolated to the host.
- You can nuke a container ..

```
rm -rf /
```

Ports can be **EXPOSED** from the container to the host. (-p)
Files can be **SHARED** to and fro the host and the container. (-v)

Ports and Files

To expose a port from a container to the host:

```
docker run -it -p 8080:80 nginx
-p   => Expose ports
8080 => Host Port
80   => Container Port
```

To share files with a container:

```
docker run -it -v /home/$(whoami):/data alpine sh
-v              => Bind Mount a Volume
/home/$(whoami) => Host Path
/data           => Container path
```

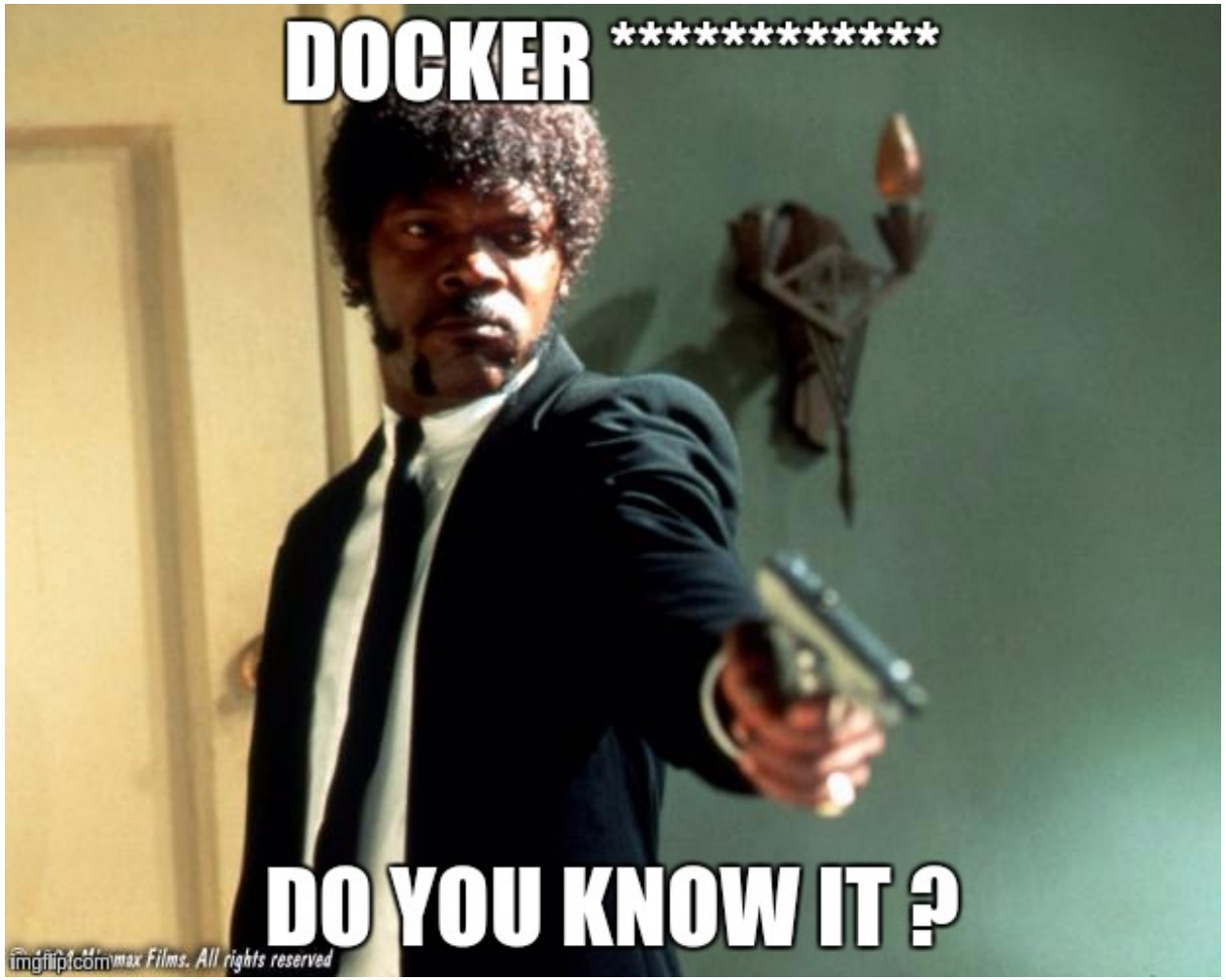
If You be like

Lets run through it again.

Checklist

- docker run -it img cmd [runs interactivley] CTRL-P + CTRL-Q = detach
- docker run -d ... [run in background]
- docker ps [show running containers]
- docker ps -a [show all i.e. exited also]
- docker rm ... [remove stopped containers]
- docker run .. -p $HOSTP:$CONTP ... [expose ports]
- docker run .. -v $HPATH:$CPATH ... [share files] **Bind** mount

Do you ... ?


DOCKER **********
DO YOU KNOW IT ?
imgflip.com max Films. All rights reserved

Lets go anonymous

```
docker run -d -p 9050:9050 -d mfrw/tor
```

Tor exposes a SOCKS listener @ 9050
We expose the port from container to our host
Configure our browser

```
-p 9050:9050 => expose container port to 9050 of host
-d           => detached execution
```

- docker ps
- docker stop ...
- docker rm ...

Just for demo purpose, it runs an ancient tor version not recommended
Make your own ;)

What happens when you install something on the container

- It remains in the container
- The image is not affected
- You have to commit the changes and make a new image

```
docker commit -t mfrw/tor:v2 #hash
```

Dockerfiles

Q: How to create a docker image ?
A: Simple Use **Dockerfile** or commit a container.

Q: So WTH is a Dockerfile ?
A: Heard of makefiles.. Something similar

Q: Can I see one ?
A: The tor one looks like this.

```
FROM alpine:latest
LABEL maintainer "Muhammad Falak R Wani <falakreyaz@gmail.com>"
RUN apk --update add tor && adduser -D anon
COPY torrc /etc/tor/torrc
EXPOSE 9050 9051
USER anon
CMD [ "tor" ]
```

Dockerfiles

Q: How do you build an image from a Dockerfile ?
A: Simple ...

```
docker build -t mfrw/tor:latest .
```

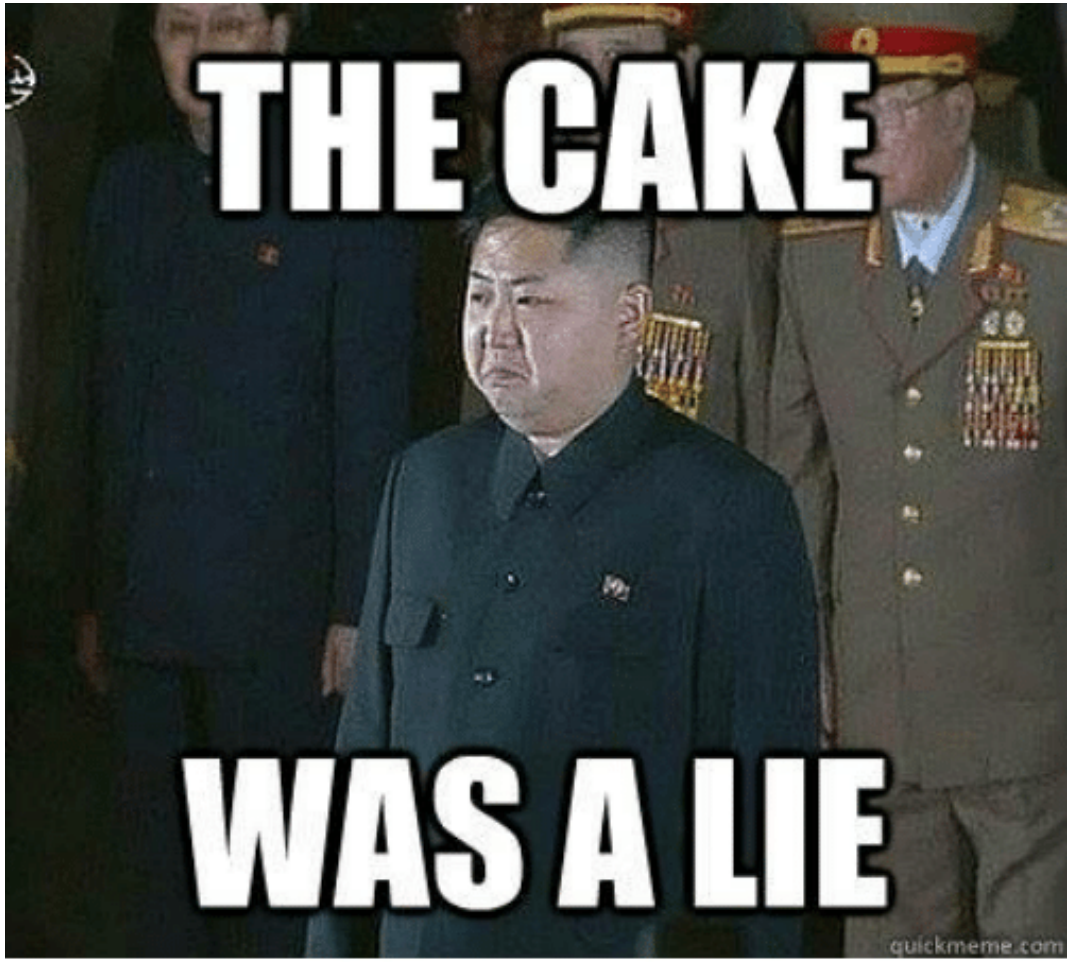
Q: Why should I use it ?
A: Its the same thing as your docker image, but easier to email ... ;)

Q: Is there a **github** lookalike for docker ?
A: **dockerhub** ... Signup

Q: Is this session over.. Im kinda bored/tired
A: The fun part begins now, we use whatever we learned till now to do some unholy stuff. You can pretty much leave if you want.

For those who chose to stay

- There is no fun part.. I **lied** ...



Dockerfile for busybox

- Remember **Bill Joy**... Yeah the **choot** dude.
- Lets write a Dockerfile for busybox.

```
FROM scratch
LABEL maintainer "Muhammad Falak R Wani <falakreyaz@gmail.com>"
COPY bb-build/bin /bin
COPY bb-build/sbin /sbin
COPY bb-build/usr /usr
COPY bb-build/linuxrc /linuxrc

CMD [ "sh" ]
```

Building ..

```
docker build -t mfrw/busybox .
```

Running ..

```
docker run -it mfrw/busybox
```

Any goT fans ?

- Lets write a small webserver in **go**
- Containarize it
- Host GOTS07E07 over it.

```
FROM scratch
LABEL maintainer "Muhammad Falak R Wani <falakreyaz@gmail.com>"
COPY fileserver /
EXPOSE 8080
CMD [ "/fileserver", "-path=/data", "-port=8080" ]
```

To run it

```
docker run -d -p 80:8080 -v $(pwd):/data mfrw/fileserver
```

gem5 (Comp-Arch CPU emulator)

```
FROM ubuntu:latest
LABEL maintainer "Muhammad Falak R Wani <falakreyaz@gmail.com>"
# get dependencies
RUN apt-get update
RUN apt-get install -y build-essential git-core m4 scons nano zlib1g zlib1g-dev libprotobuf-dev protobuf
# checkout repo with mercurial
RUN useradd -u 1000 -ms /bin/bash gem5 && chown -R gem5:gem5 /home/gem5
WORKDIR /home/gem5
# build it
WORKDIR /home/gem5
USER gem5
RUN git clone --depth 1 https://github.com/gem5/gem5.git
RUN cd gem5 && scons -j$(nproc) --ignore-style build/X86/gem5.opt
USER root
RUN apt-get clean
RUN apt-get purge --auto-remove -y \
    && rm -rf /var/lib/apt/lists/* \
    && rm -rf /src/*.deb
USER gem5
ENTRYPOINT bash
```

Languages

node.js

```
docker run -it -w /data -v $PWD:/data node
```

golang

```
docker run -it -w /data -v $PWD:/data golang
```

ruby

```
docker run -it -w /data -v $PWD:/data ruby
```

django

```
docker run -it --net=host -w /data -v $PWD:/data django
```

How About Google-Chrome ?

```
FROM ubuntu:latest
LABEL maintainer "Muhammad Falak R Wani <falakreyaz@gmail.com>"
ADD https://dl.google.com/linux/direct/google-talkplugin_current_amd64.deb /src/google-talkplugin_curren
RUN apt-get update && apt-get install -y apt-transport-https ca-certificates \
    curl gnupg hicolor-icon-theme libgl1-mesa-dri libgl1-mesa-glx \
    libpango1.0-0 libpulse0 libv4l-0 fonts-symbola --no-install-recommends \
    && curl -sSL https://dl.google.com/linux/linux_signing_key.pub | apt-key add - \
    && echo "deb [arch=amd64] https://dl.google.com/linux/chrome/deb/ stable main" > /etc/apt/sources.li
    && apt-get update && apt-get install -y google-chrome-stable --no-install-recommends \
    && dpkg -i '/src/google-talkplugin_current_amd64.deb' && apt-get purge --auto-remove -y curl \
    && rm -rf /var/lib/apt/lists/* && rm -rf /src/*.deb
RUN groupadd -r chrome && useradd -r -g chrome -G audio,video chrome \
    && mkdir -p /home/chrome/Downloads && chown -R chrome:chrome /home/chrome
RUN rm -rf /var/cache/apt/archives/*
ENV PULSE_SERVER /home/chrome/pulse
COPY local.conf /etc/fonts/local.conf
USER chrome
CMD [ "google-chrome" ]
```

Dockerfiles can be very messy.

Google-Chrome

Run it :

```
docker run --rm -it \
        -v /tmp/.X11-unix:/tmp/.X11-unix \
        -e DISPLAY=$DISPLAY --privileged \
        -v /dev/shm:/dev/shm --name chrome \
        mfrw/chrome
```

Add Sound :

```
docker run --rm -it \
        -v /run/user/1000/pulse/native:/home/chrome/pulse \
        -v /tmp/.X11-unix:/tmp/.X11-unix \
        -e DISPLAY=$DISPLAY --privileged \
        -v /dev/shm:/dev/shm --name chrome \
        mfrw/chrome
```

If it doesn't work, try after running:

```
xhost +
```

Latex Compiler

```
#!/bin/bash
docker container run -it --rm \
    -v $PWD:/data \     # Mount Host $PWD to Container /data
    -h latex \        # hostname
    -u  1000 \        # uid
    --name latex \
    mfrw/latex
```

Docker on steroids

Host Full sharelatex.com locally
In under 5 minutes.

```
docker-compose up
```

You can use it also if you want :)

```
IP       : 192.168.1.41
Username : test@iiitd.ac.in
Password : abc123
```

Questions ?

Feedback

- I would be honored if you gave me honest feedback.
- I am happy to answer a question after you **RTFM**.
- If you want more, I almost always reply to emails.

Thank you

Muhammad Falak R Wani
https://github.com/mfrw/talks/tree/master/docker-intro (https://github.com/mfrw/talks/tree/master/docker-intro)
falak16018@iiitd.ac.in (mailto:falak16018@iiitd.ac.in)