EMBRACE TechTalks

- 24 . 04 . 2019 -

docker

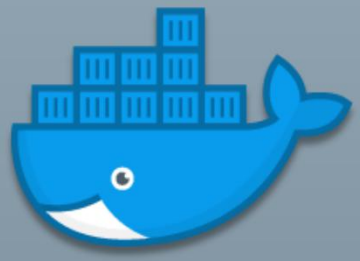
David Guimarães

https://github.com/daviguima
david.franca@inpe.br
dvdgmf@gmail.com

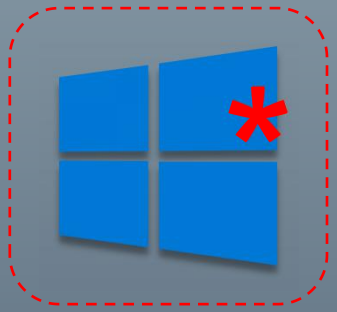
WHAT THE censored IS DOCKER!?

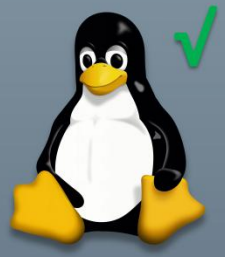
- OS-level virtualization
- 2013
- Containers
- C x C = isolated
- C = individual libs & conf

O.S. AGNOSTIC

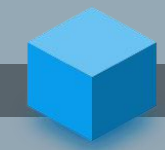


N ...





*https://success.docker.com/article/compatibility-matrix



BUT WHY IS IT SO TRENDY?

[...] Docker is a computer program that performs operating-system-level virtualization.[6] It was first released in 2013 and is developed by Docker, Inc.[7]

Docker is used to run software packages called containers. Containers are isolated from each other and bundle their own application,[8] tools, libraries and configuration files; **they can communicate with each other** through well-defined channels. **All containers are run by a single operating-system kernel and are thus more lightweight than virtual machines**. Containers are created from images that specify their precise contents. Images are often created by combining and modifying standard images downloaded from public repositories. ***[...]***

WHO USES?

Google Cloud

PayPal

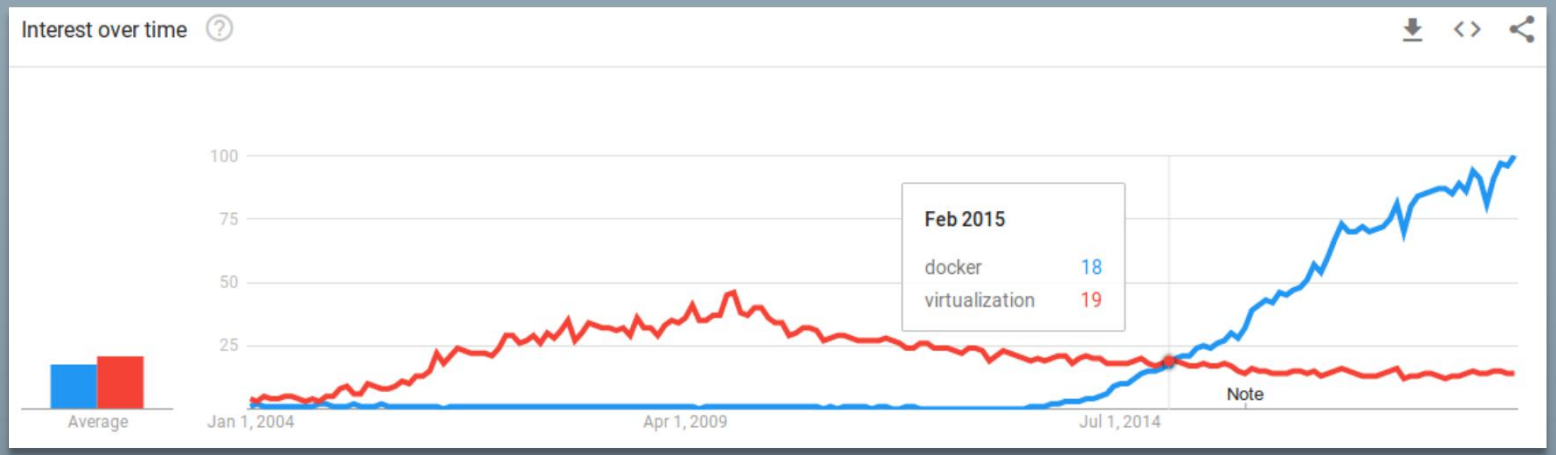
amazon web services

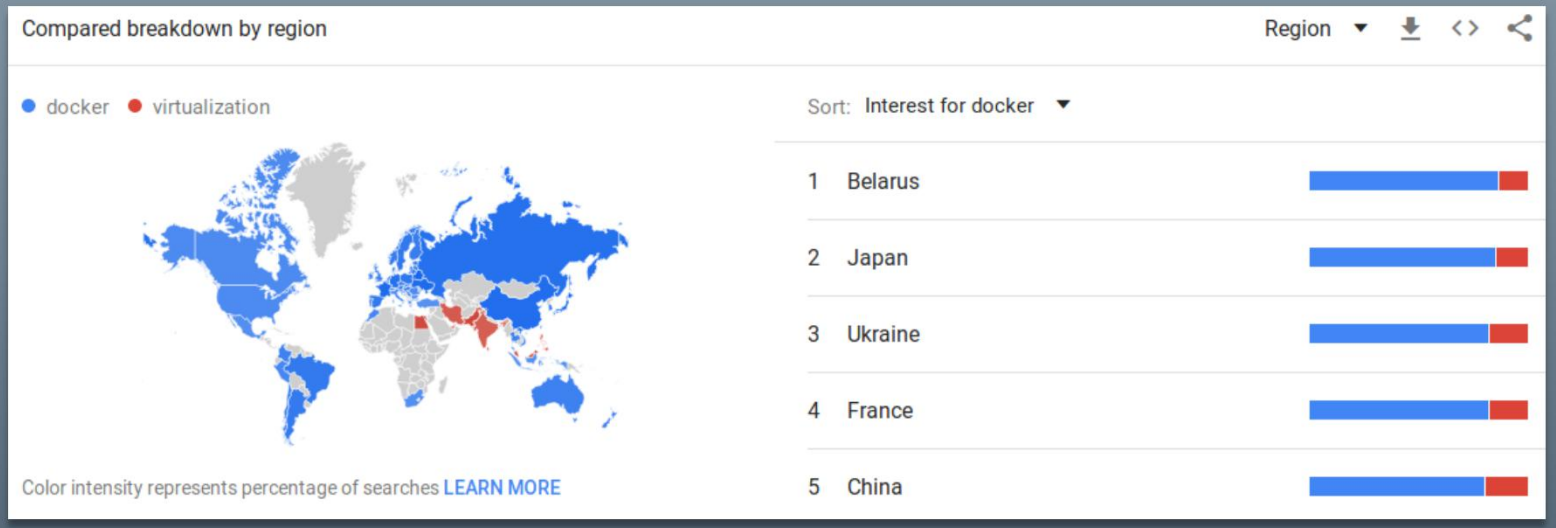
VISA

Microsoft Azure

WHO CARES?

Google Trends





SOURCE: https://trends.google.com/trends/explore?cat=5&date=all&q=docker,virtualization

BENCHMARK

kubernetes rkt

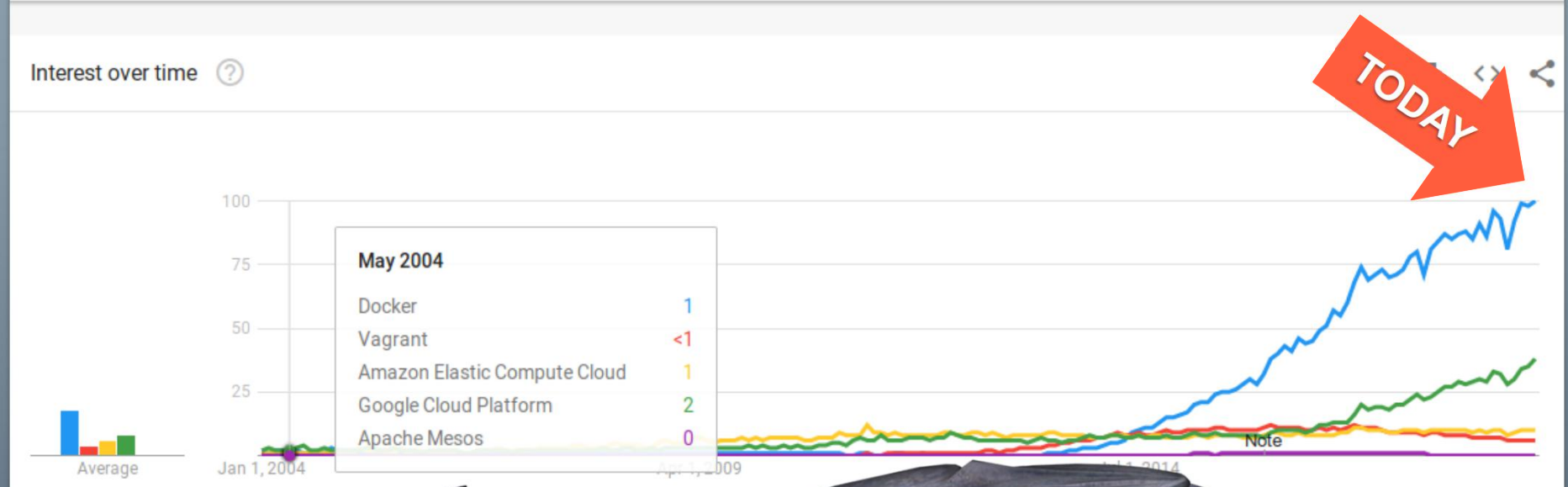
amazon web services

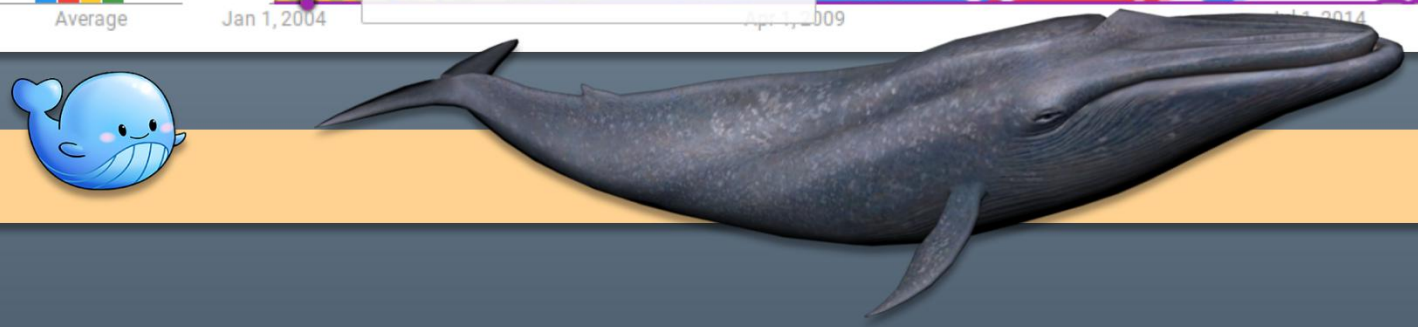
MESOS

Docker	Vagrant	Amazon Elastic Co ...	Google Cloud Platf ...	Apache Mesos
Software	Software	Computer application	Computer application	Topic

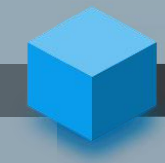
Worldwide ▾ 2004 - present ▾ Computers & Electronics ▾ Web Search ▾

Interest over time





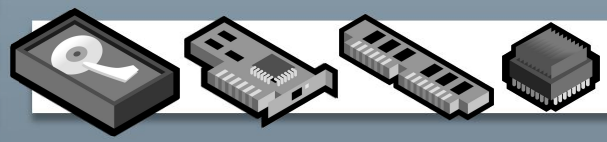
VIRTUALIZATION LAYERS



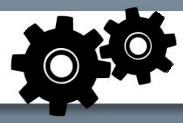
{APP-1} {APP-2} {APP-3}

bins / libs bins / libs bins / libs

Guest OS



Virtual HW



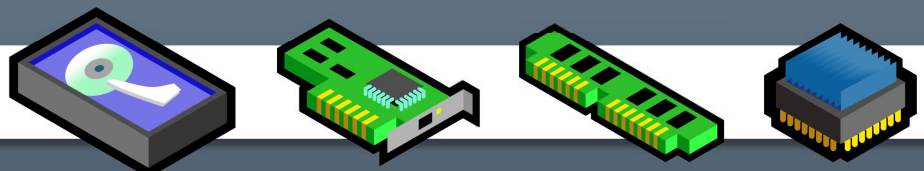
Hypervisors

{APP-1} {APP-2} {APP-3}

C1 bins / libs C2 bins / libs C3 bins / libs

Docker Virtualization Layer

HOST O.S. Kernel



Physical Host

VIRTUALIZATION LAYERS

700 MB * 3 = **2.1** GB
+ CPU and memory resources

{APP-1} **{APP-2}** **{APP-3}**

C1 bins / libs | **C2** bins / libs | **C3** bins / libs

Docker Virtualization Layer

{APP-1} **{APP-2}** **{APP-3}**

bins / libs | bins / libs | bins / libs

Guest OS

Virtual HW

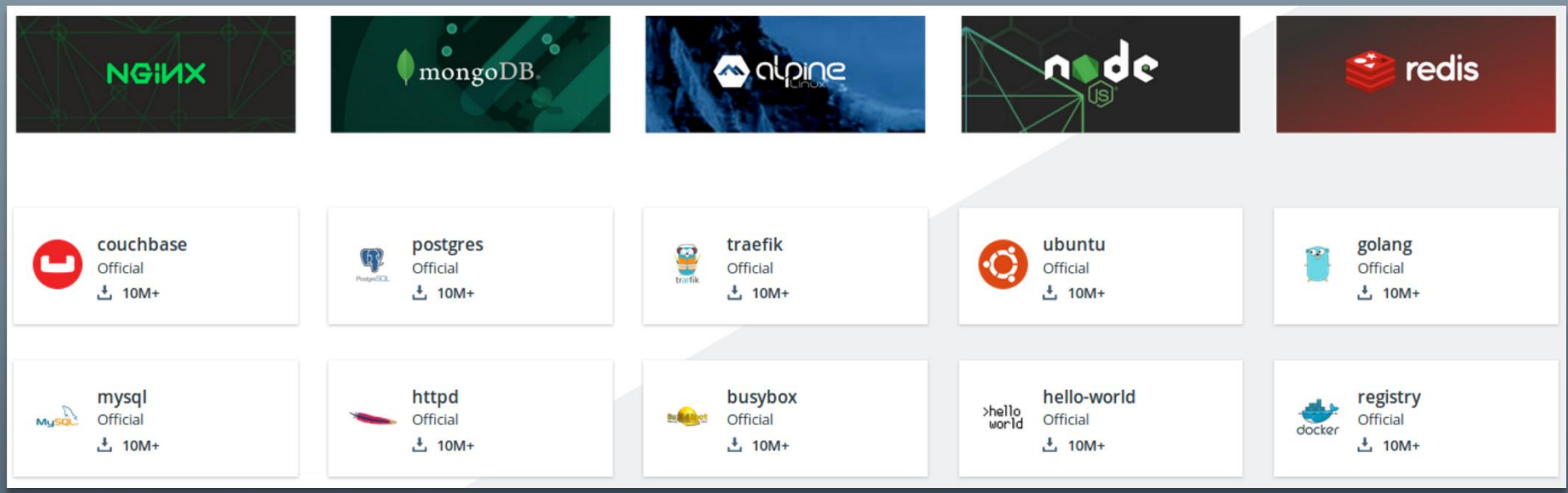
Hypervisors

HOST O.S. Kernel

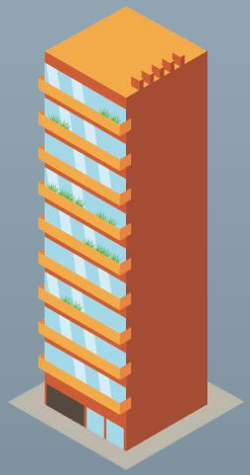
Physical Host

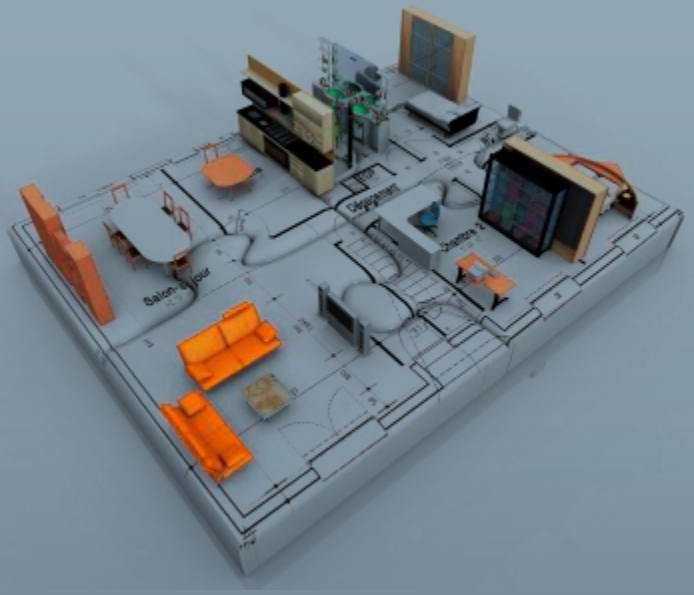
DOCKER HUB

GitHub ⟷ docker HUB



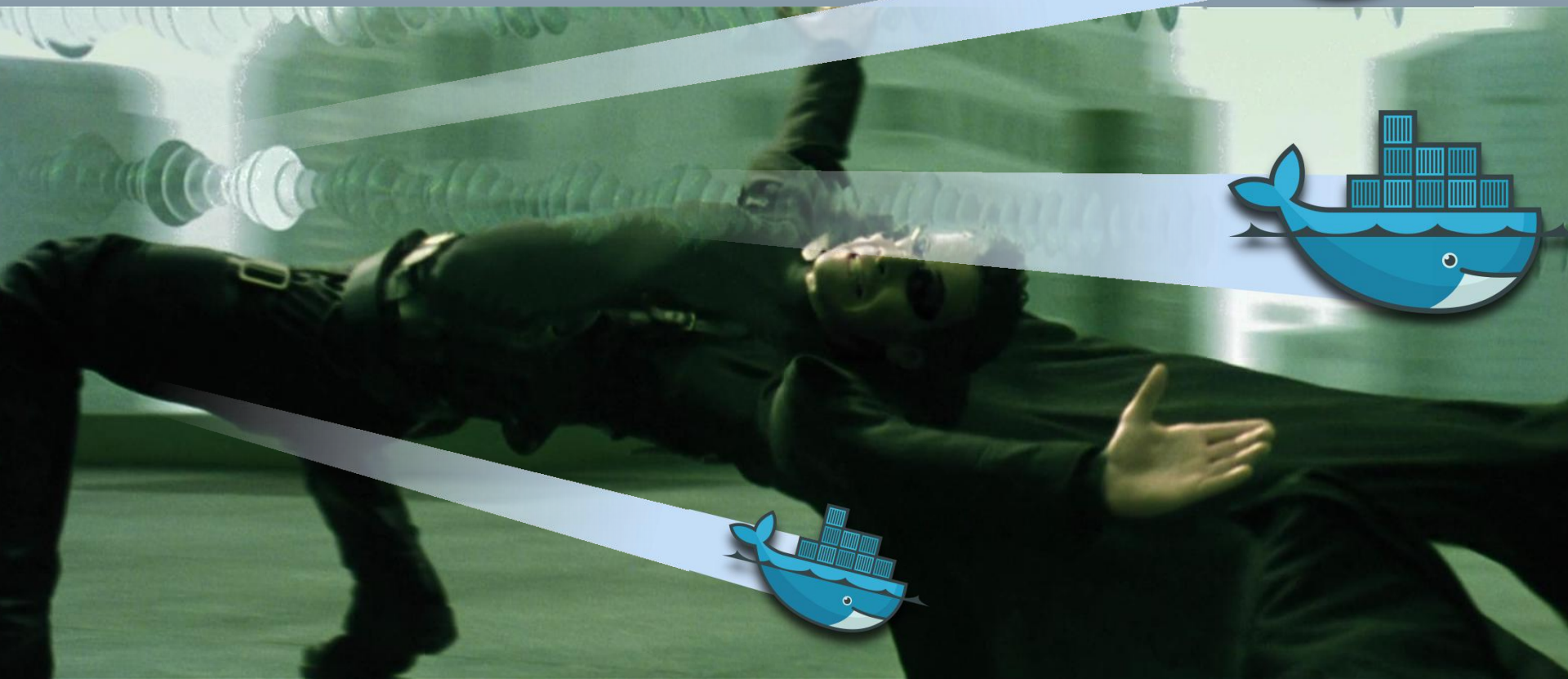
DEFINITIONS

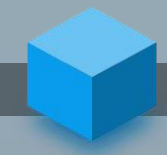




CONTAINER IMAGE ≠ **CONTAINER** ≠ **VIRTUAL MACHINE**

HANDS ON!





REPOSITORY SETUP

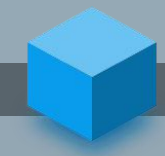
```
# Update the apt package index:
$ sudo apt update

# Install packages to allow apt to use a repository over HTTPS:
$ sudo apt install apt-transport-https ca-certificates \
curl gnupg-agent software-properties-common

# Add Docker’s official GPG key:
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

# Verify that you now have the key with the fingerprint
$ sudo apt-key fingerprint 0EBFCD88

pub   rsa4096 2017-02-22 [SCEA]
      9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
uid           [ unknown] Docker Release (CE deb) <docker@docker.com>
sub   rsa4096 2017-02-22 [S]

# Setup the stable repository (x86_64/amd64):
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
          $(lsb_release -cs) stable"

# Install latest Docker CE
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

REPOSITORY SETUP

```
# Verify that Docker CE is installed correctly:
$ sudo docker run hello-world



Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:92695bc579f31df7a63da6922075d0666e565ceccad16b59c3374d2cf4e8e50e
Status: Downloaded newer image for hello-world:latest
```

REPOSITORY SETUP

```
# Verify that Docker CE is installed correctly:
$ sudo docker run hello-world

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/
```



REPOSITORY SETUP

TIME TO GET DIRTY.

```
$ git clone git@github.com:daviguima/nginx_fun.git
```

>>>

REPOSITORY SETUP

```
# Thats all for now folks.
$ THANK YOU!!! :)
```


David Guimarães
https://github.com/daviguima
david.franca@inpe.br
dvdgmf@gmail.com

REPOSITORY SETUP

```
# Thats all for now folks.
$ THANK YOU!!! :)
```


David Guimarães
https://github.com/daviguima
david.franca@inpe.br
dvdgmf@gmail.com

3EGWeRTojJQrufnQs9CDLJ6R8NgpBBdFfk