Docker on Google Cloud Platform

What is Docker?
Why is it important for the Cloud?

Deploy a Java 8 webapp
With Docker and Google Cloud Platform

David Gageot

http://javabien.net

@dgageot

http://www.flickr.com/photos/fortcarson/

And you
Docker? App Engine? Compute Engine?

What is Docker?

What's a VM

Sometimes worse

Different Operating System

Different CPU Architecture

Works on my machine!

Work with multiples versions in //

Move application to a different server

...

What for?

" Everything at Google, from Search to Gmail, is packaged and run in a Linux container.

Each week we launch more than 2 billion container instances across our global data centers, and the power of containers has enabled both more reliable services and higher, more-efficient scalability. "

http://googlecloudplatform.blogspot.fr/2014/06/an-update-on-container-support-on-google-cloud-platform.html

What Google does

Demo of a simple container

```
$ docker run -i -t ubuntu:14.04 /bin/bash

$ uname -a

$ ls -als /
```

Use an existing container

```
$ docker run -i -t ubuntu:14.04 /bin/bash

$ sudo rm -Rf /etc
$ ls /etc

$ docker run -i -t ubuntu:14.04 /bin/bash

$ ls -als /
$ exit
```

Fun with the Container

Size
Disk & RAM

State (**Memory, Written Data**)

Application (**Jars**)

Binaries and Libraries (**JDK**)

Boot
Time

Common Files (**/etc, /bin,…**)

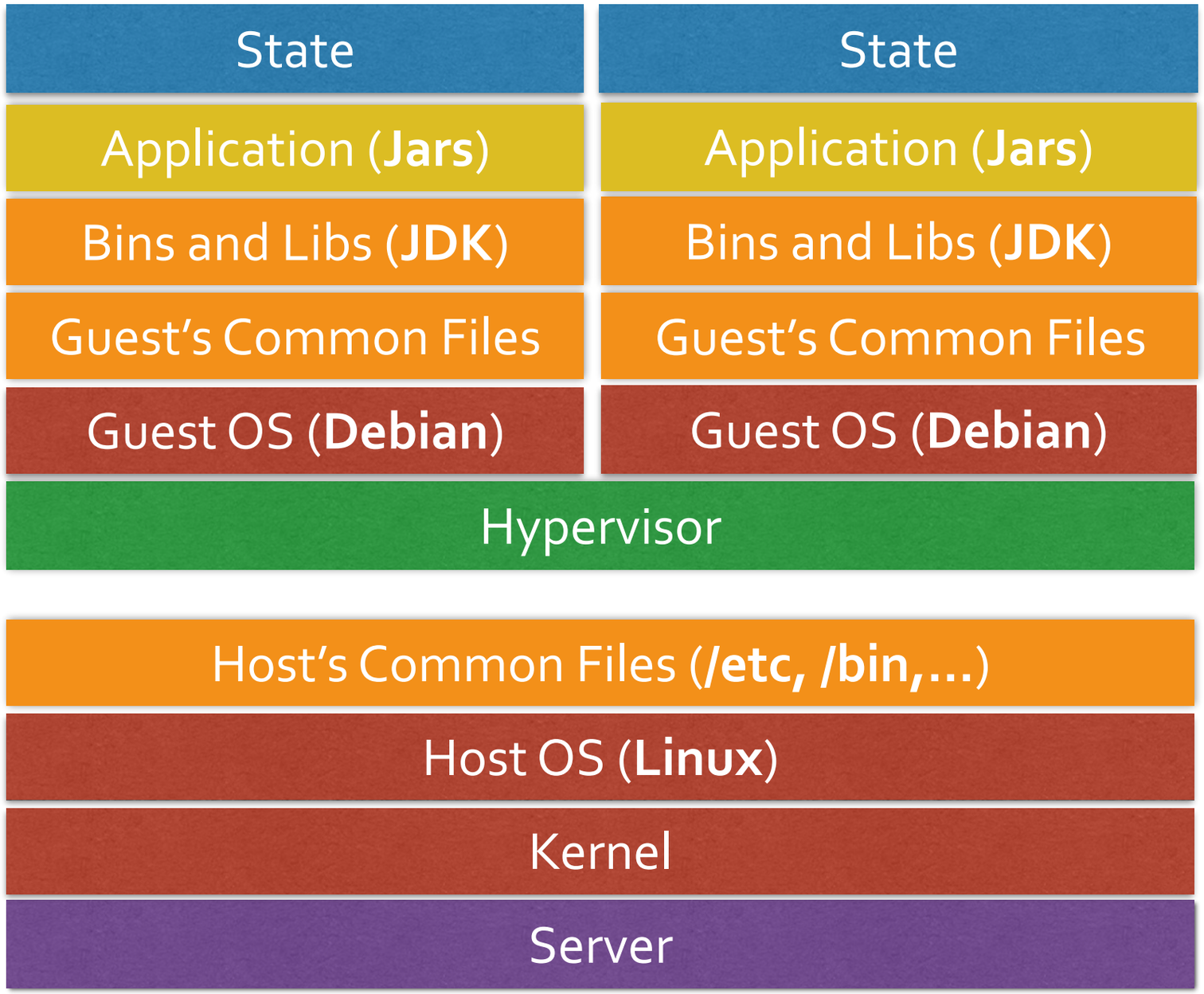
OS (**Linux**)

Kernel

Server

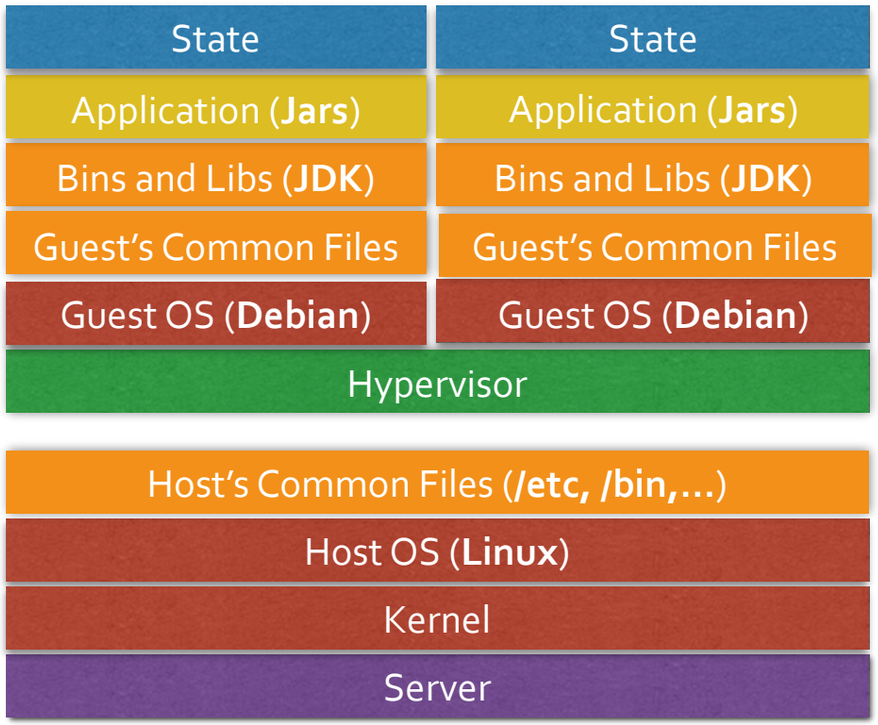
What's an application?

Size
Disk & RAM



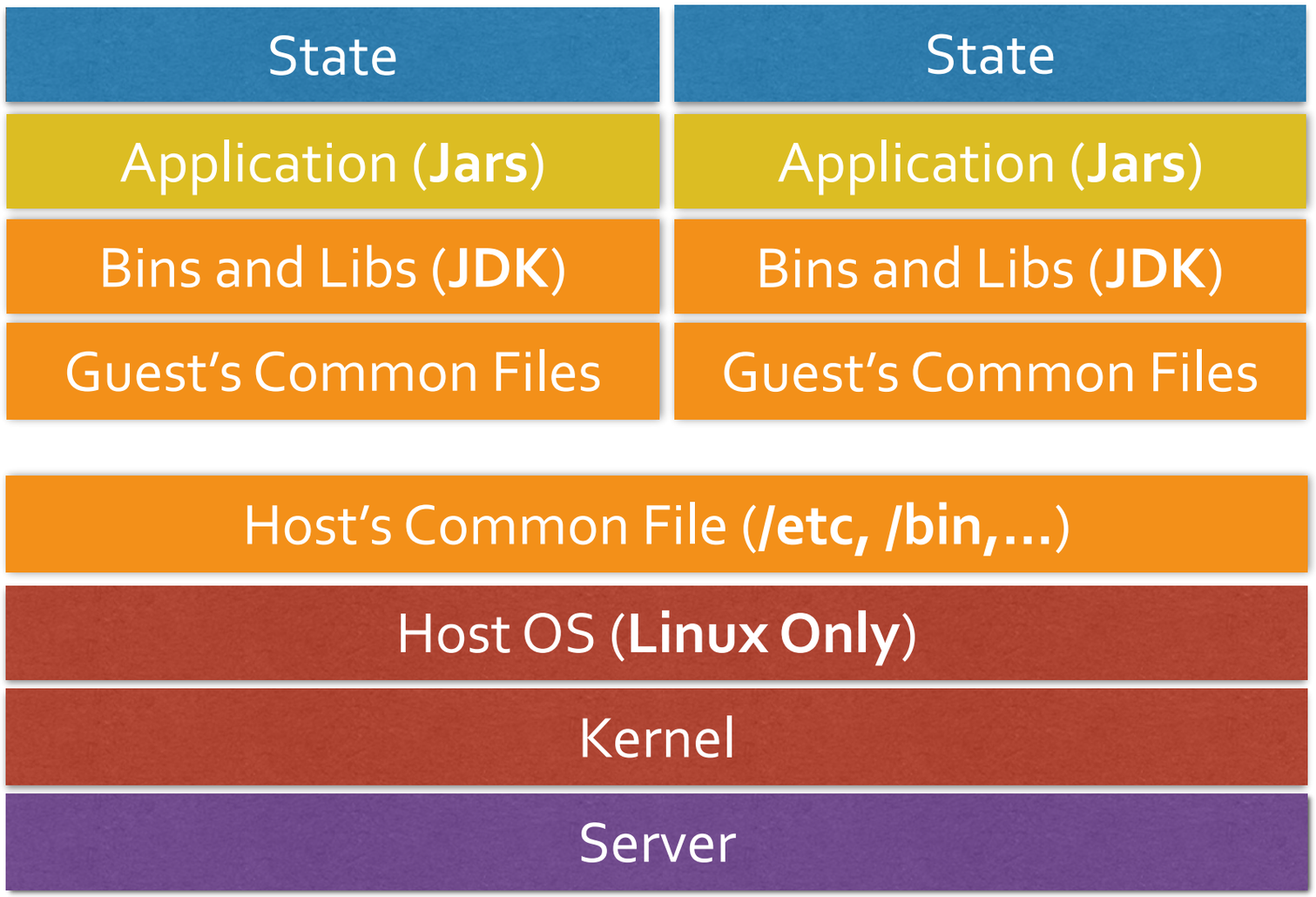
Boot
Time

And in a VM?

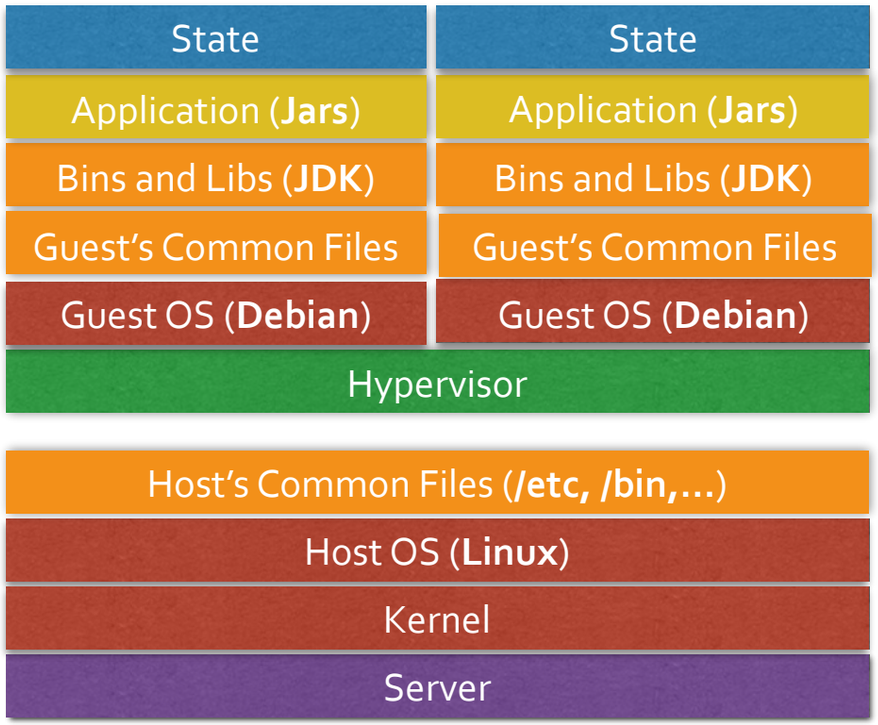


VMs

docker

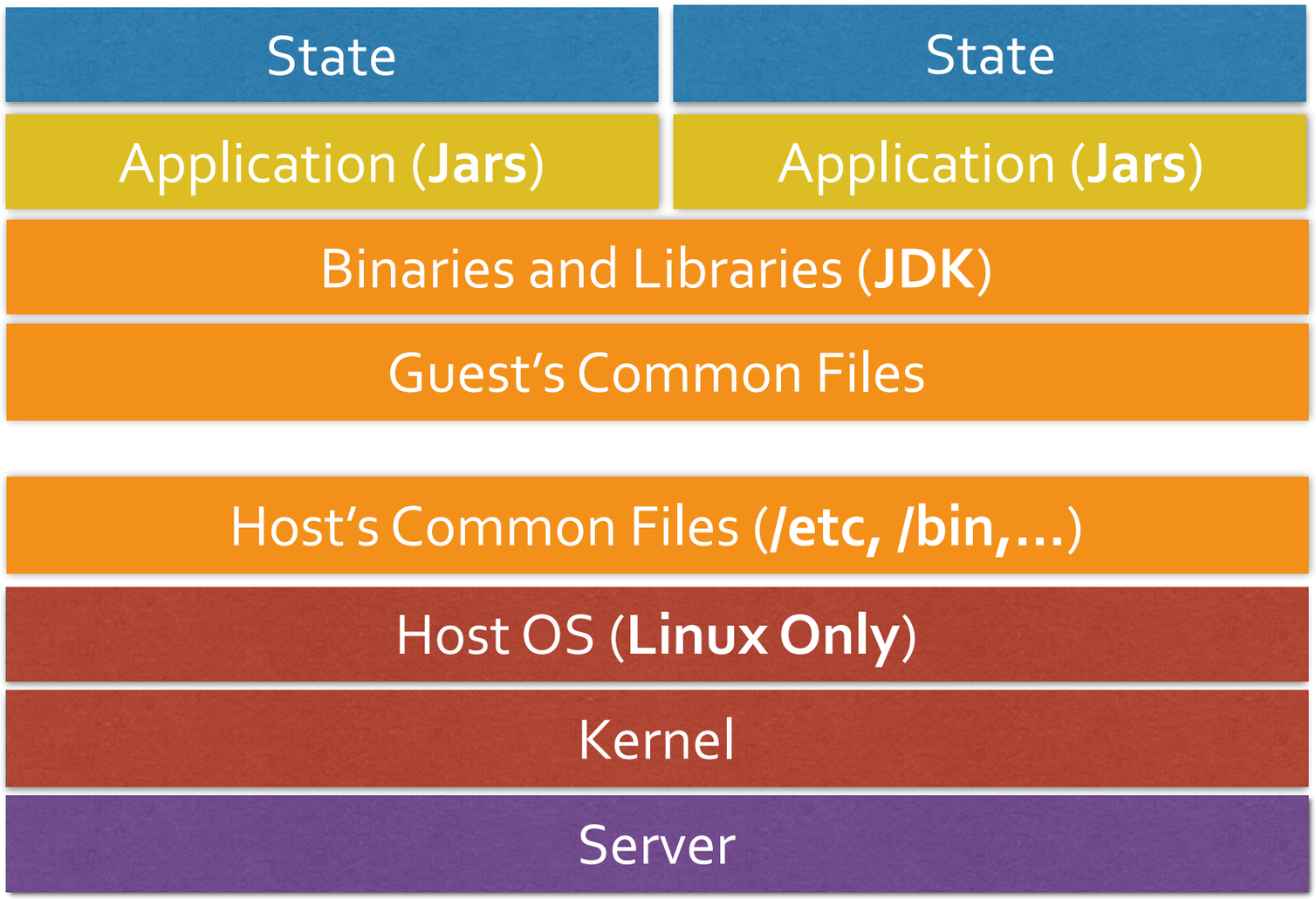


One Kernel to rule them all

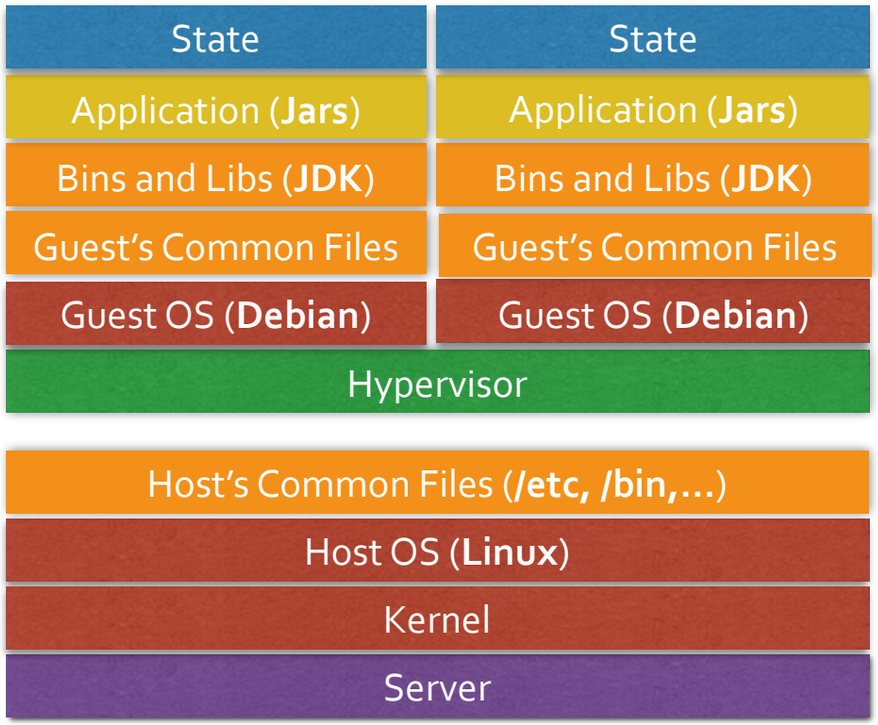


VMs

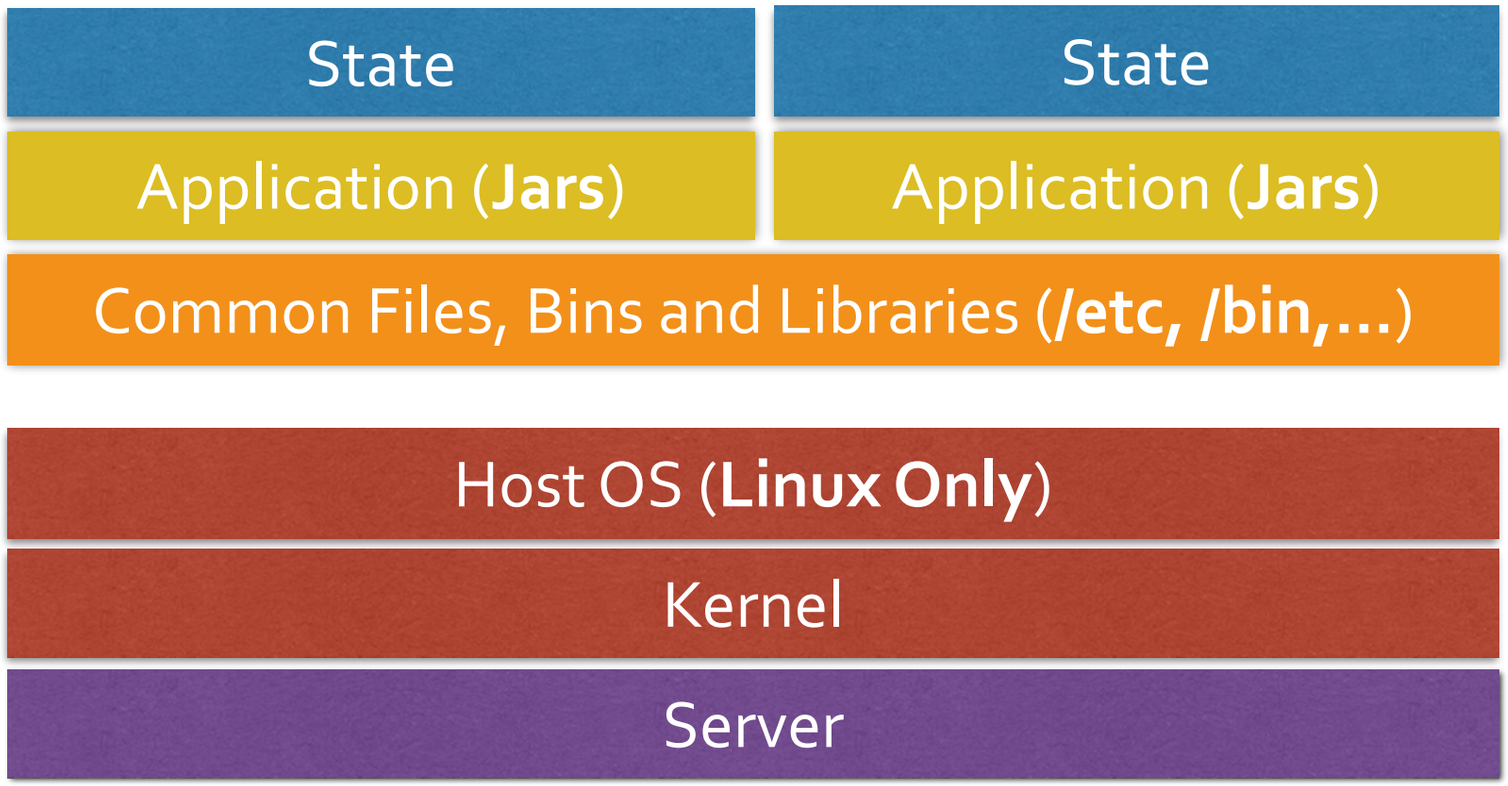
docker



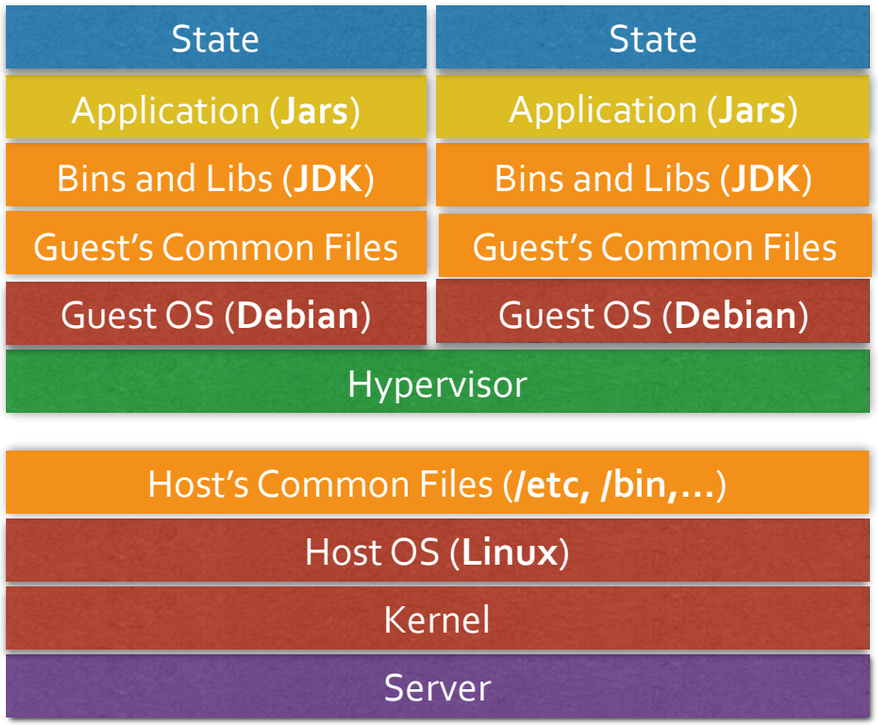
Share the Read only files



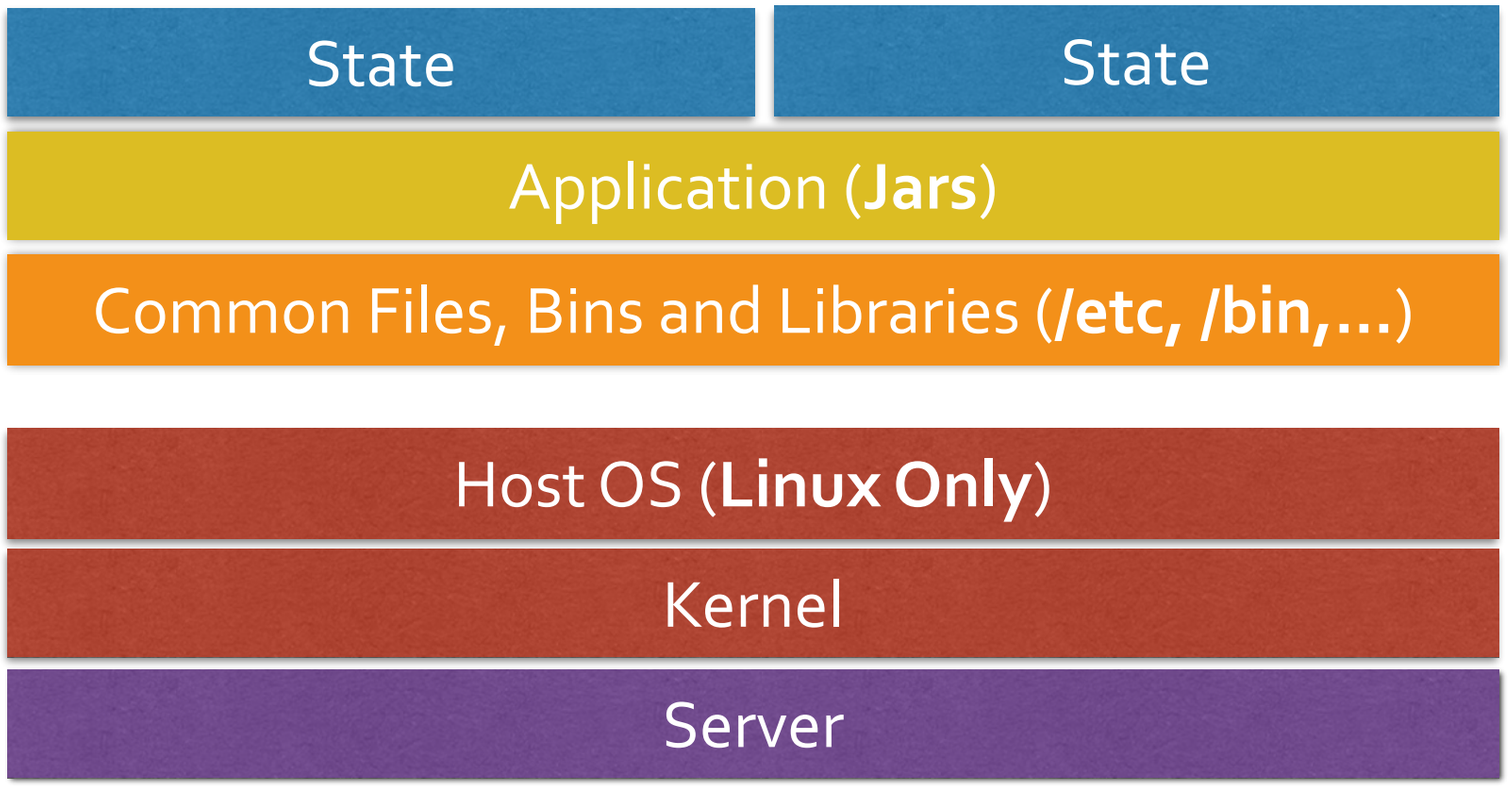
VMs



It's all about files

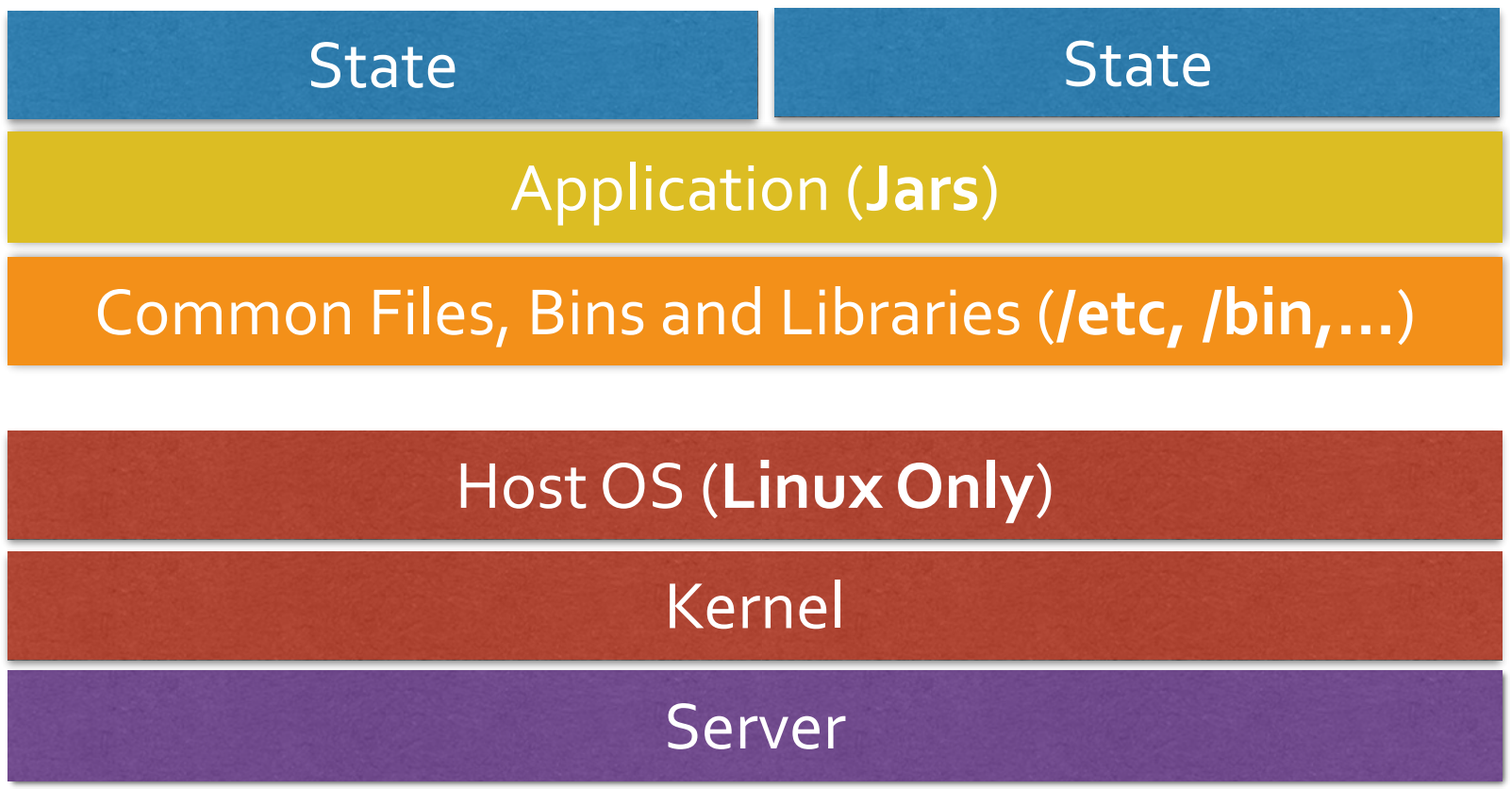


VMs



Share the Application

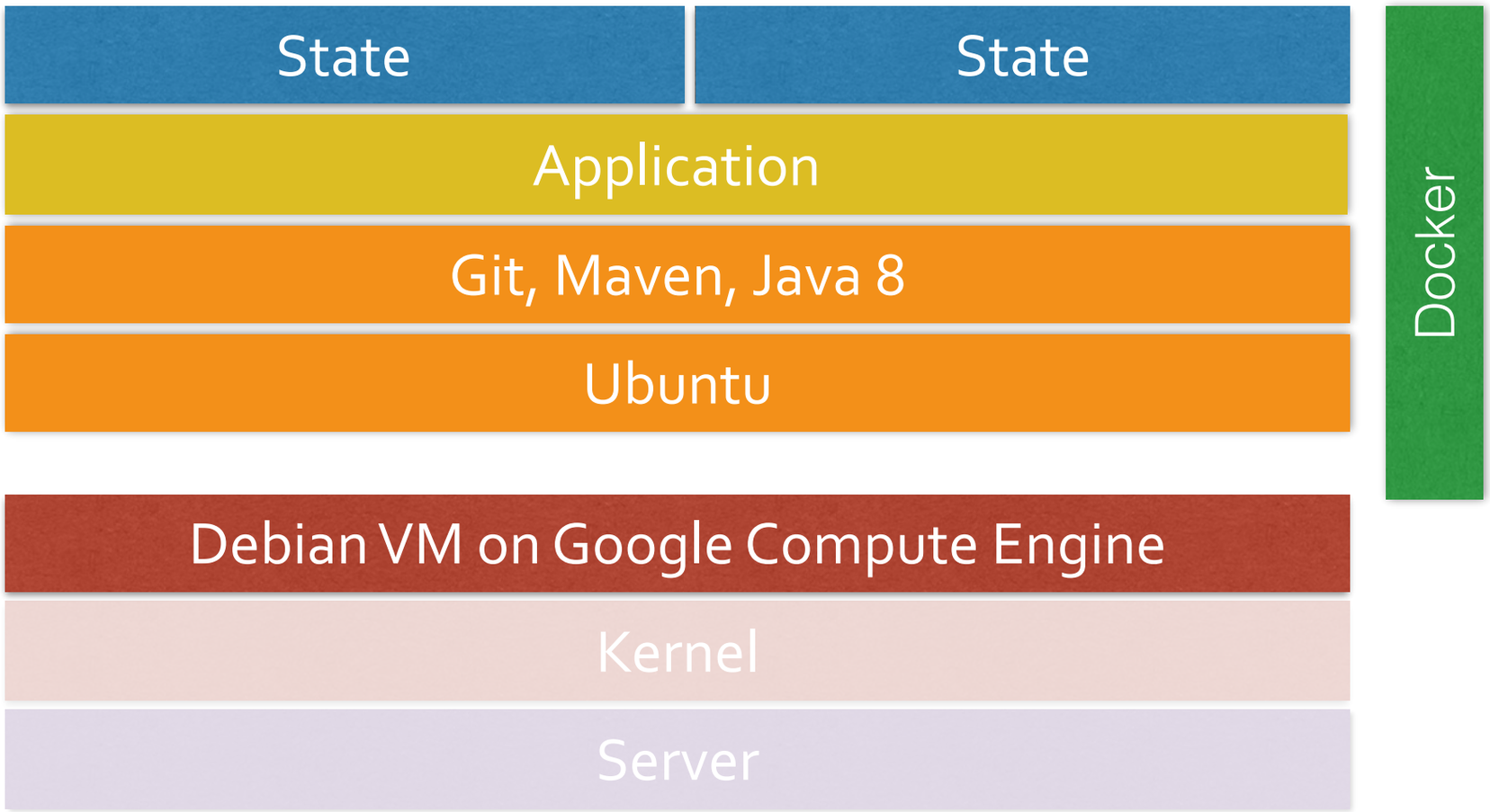
Size
Disk & RAM



Boot
Time

As fast as “native” apps

It's a Container



Our target Webapp

Package a Java 8 webapp

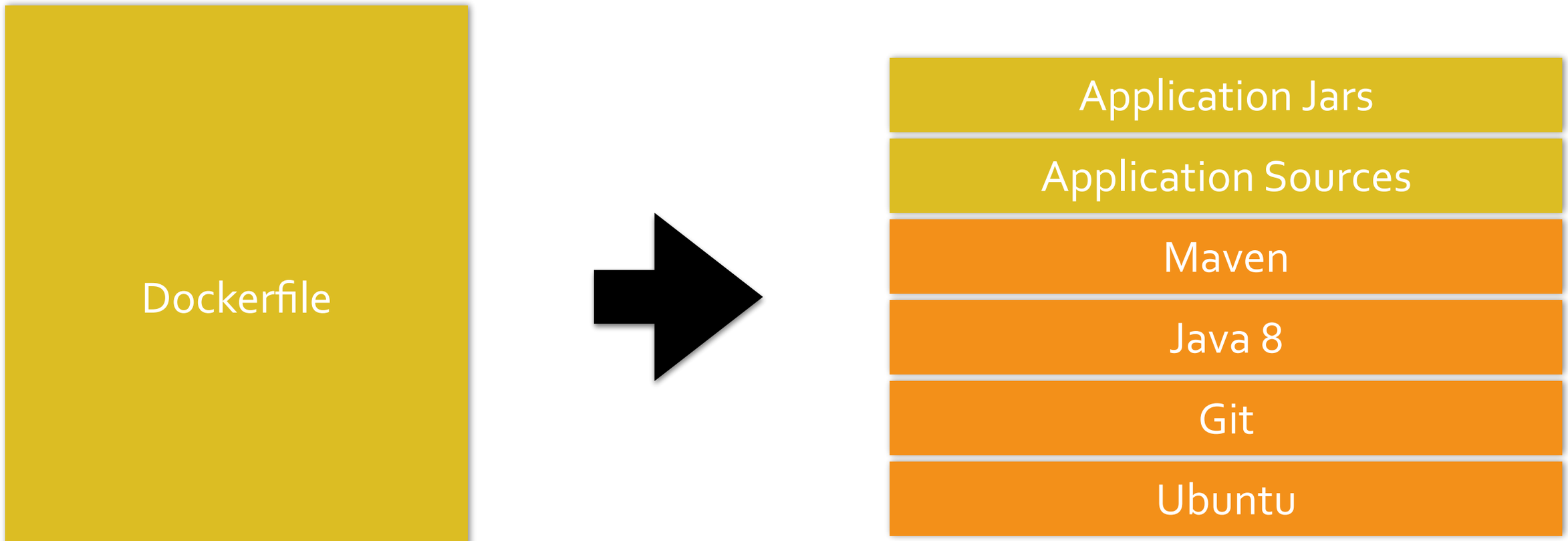
```
from base
maintainer David Gageot <david@gageot.net>

# Install prerequisites
run apt-get update
run apt-get install -y software-properties-common

# Install java8
run add-apt-repository -y ppa:webupd8team/java
run apt-get update
run echo oracle-java8-installer shared/accepted-oracle-license-v1-1 select true | sudo /usr/bin/debconf-set-selections
run apt-get install -y oracle-java8-installer

# Install tools
run apt-get install -y git maven

# Clone project
run git clone https://github.com/dgageot/helloworld.git

# Build project
run cd helloworld && mvn verify dependency:copy-dependencies

# Expose the http port
expose 8080

workdir helloworld
```

ie ubuntu

Describe your application with a Dockerfile

This one is hosted on GitHub

$ docker build -t dgageot/helloworld github.com/dgageot/helloworld



Build the Container

Each command creates a new read-only layer

$ docker build -t dgageot/helloworld github.com/dgageot/helloworld
$ docker build -t dgageot/helloworld github.com/dgageot/helloworld

```
Uploading context 190.5 kB
Uploading context
Step 0 : from base
 ---> b750fe79269d
Step 1 : maintainer David Gageot <david@gageot.net>
 ---> Using cache
 ---> c50bcc57f807
Step 2 : run apt-get update
 ---> Using cache
 ---> 0a3783337ecb
Step 3 : run apt-get install -y software-properties-common
 ---> Using cache
 ---> 5399953b8138
Step 4 : run add-apt-repository -y ppa:webupd8team/java

.........
```

Fun with the Container
Try to build twice and see the cache being used

Java installed on host:

$ java -version

Java installed on the container:

$ docker run dgageot/helloworld java -version

Fun with the Container
Run java from the container

```
$ docker run -p 80:8080 -t -i dgageot/helloworld \
   java -jar target/hello.jar
```

or

```
$ docker run -p 80:8080 -t -d dgageot/helloworld \
   java -jar target/hello.jar
$ docker ps
```

Fun with the Container
Run the java 8 webapp

Deploy on a VM


```
$ gcutil \
    --service_version=v1 \
    --project=numeric-scope-568 \
addinstance hello \
    --zone=europe-west1-b \
    --machine_type=n1-standard-1 \
    --tags=http-server \
    --image="https://www.googleapis.com/compute/v1/projects/
debian-cloud/global/images/backports-debian-7-wheezy-v20140415"
```

Deploy on Compute Engine
Create a server instance


```
$ gcutil \
    --service_version=v1 \
    --project=numeric-scope-568 \
ssh \
    --zone=europe-west1-b \
hello
```

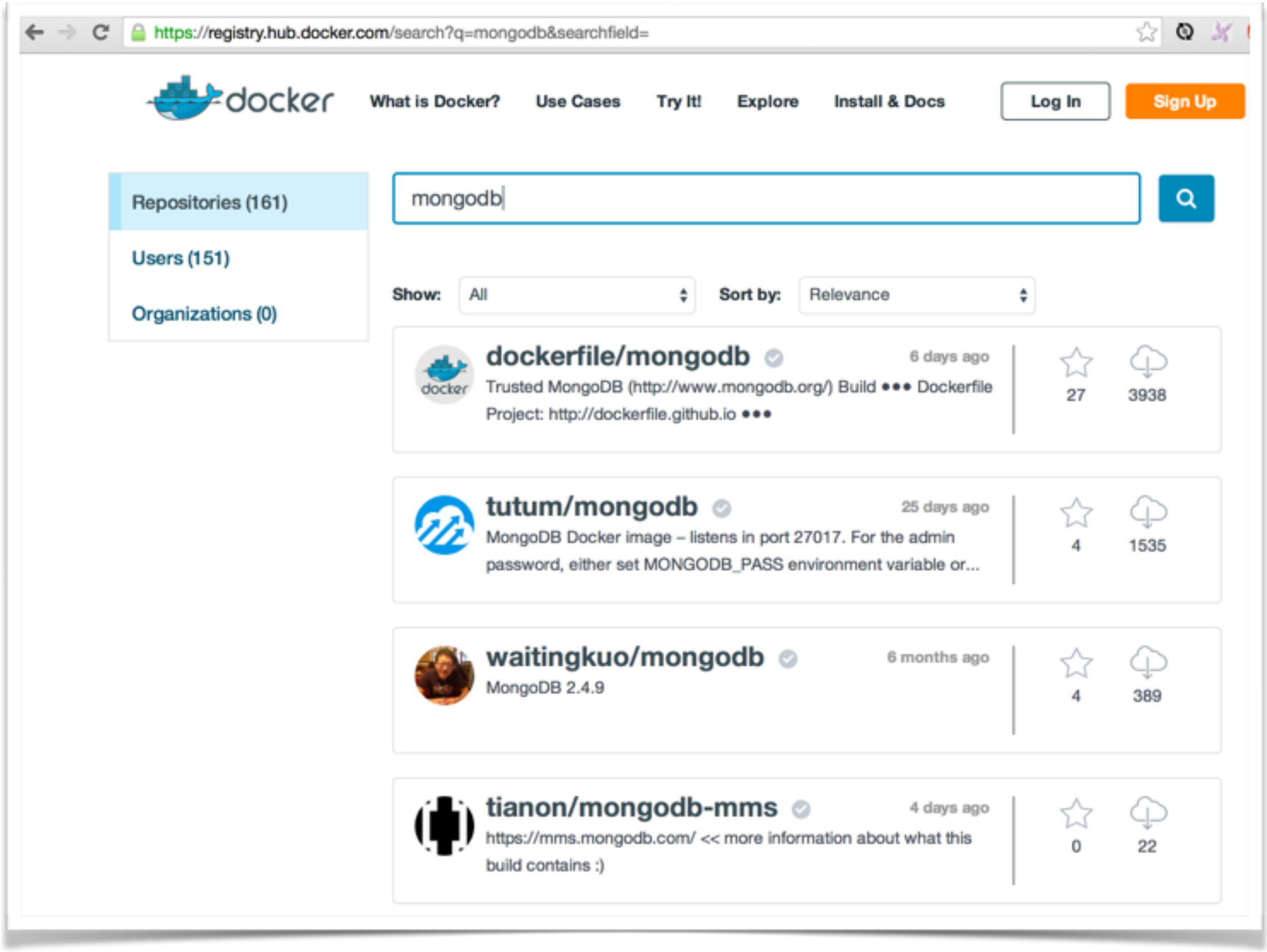
Ssh into the VM

```
$ curl get.docker.io | bash
```

Install Docker

Same thing as on localhost.
That’s the whole point!

Build and Run the container



Use a container image instead of build from Dockerfile
Need to publish the image (public or private)

Packaged as a container

docker

What is Docker? Use Cases Try It! Explore Install & Docs Log In

AUTOMATED BUILD REPOSITORY

Updated 5 days, 1 hour ago

google / docker-registry

Pull this repository docker pull google/docker-registry

No description set

☆ 3 0 604

Information | Build Details | Tags

Docker-registry

Sources for google/docker-registry, Docker Registry image to push/pull your Docker images to/from Google Cloud Storage.

- Uses 'gcs' as a storage option
- Has OAuth2 support built in
- Works locally and on [Google Compute Engine] (https://cloud.google.com/products/compute-engine/)

Build Details

Links

Source Project Page

Source Repository

Files

Build Bundle

Dockerfile

Manage

google/docker-registry
Proxy to private repos on Cloud Storage

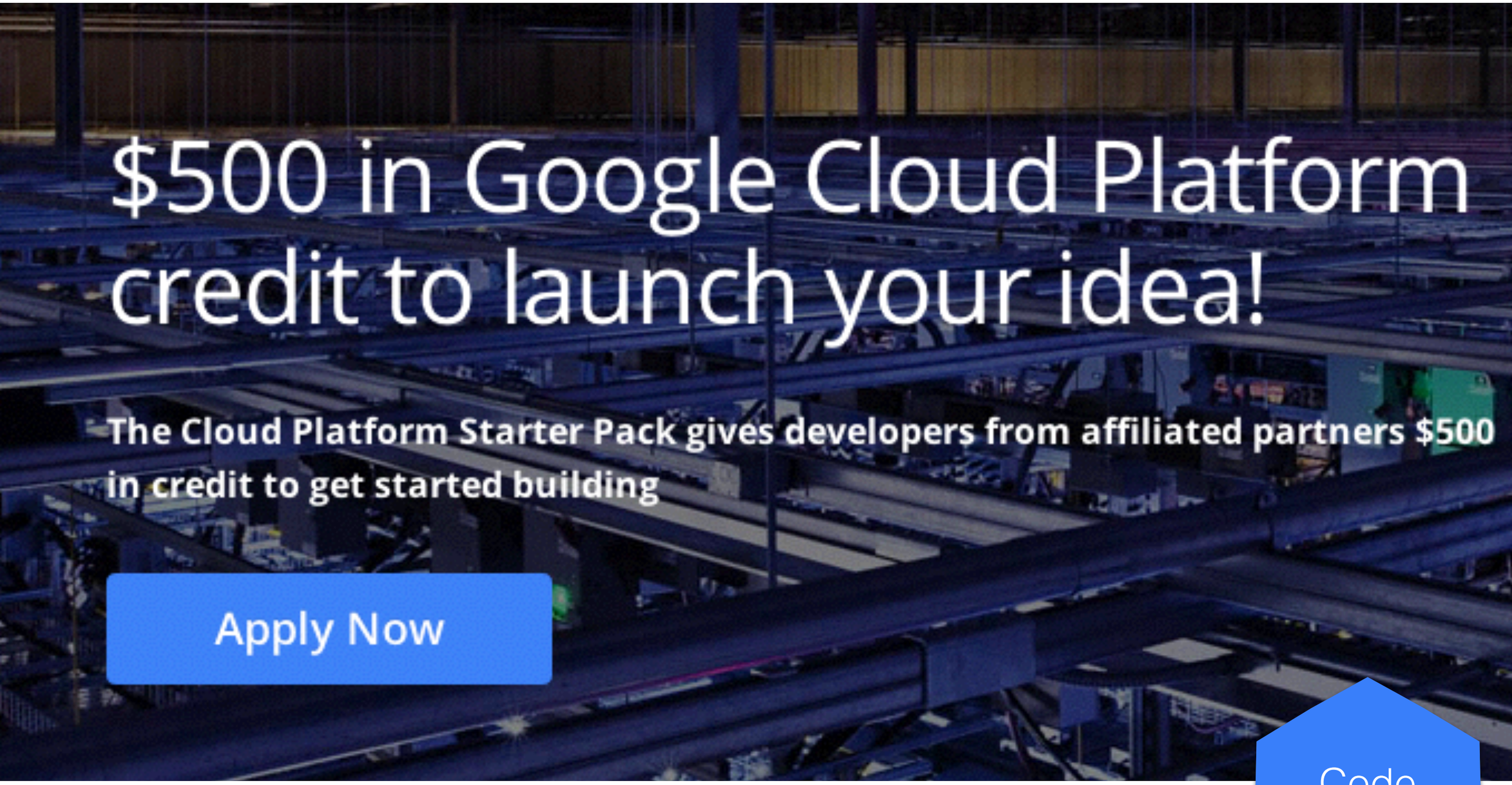
Deploy as a Container


```
$ cat containers.yaml

version: v1beta1
containers:
  - name: helloworld
    image: dgageot/helloworld
    command: ['java', '-jar', 'target/hello.jar']
    ports:
      - name: http
        hostPort: 80
        containerPort: 8080
```

Describe the setup


```
gcloud compute instances create jug \
    --image projects/google-containers/global/images/container-vm-
v20140522 \
    --metadata-from-file google-container-manifest=containers.yaml \
    --zone europe-west1-b \
    --machine-type n1-standard-1 \
    --tags=http-server
```

Deploy


$500 in Google Cloud Platform credit to launch your idea!
The Cloud Platform Starter Pack gives developers from affiliated partners $500 in credit to get started building
Apply Now
Code
gde-in

Bonus

The future of Docker
On Google Cloud Platform

Support of Docker images on App Engine

Through managed VMs

Kubernetes

Kubernetes builds on top of Docker to construct a clustered container scheduling service.

cAdvisor

Fine-grain statistics on resource usage for containers

Eric Brewer, VP of Infrastructure Google

Nominated to Docker's Governance Committee

The future of Docker On Google Cloud Platform

Enfin nous déploierons ce site web sur la Google Cloud Platform en utilisant Compute Engine avec du load balancing. Ce déploiement sera aussi l'occasion de voir comment faire communiquer plusieurs Dockers entre eux.

Q&A