Docker

Massimiliano Dessi

25 nov 2015

GDG Sardegna

Speaker @desmax74

GDG Sardegna

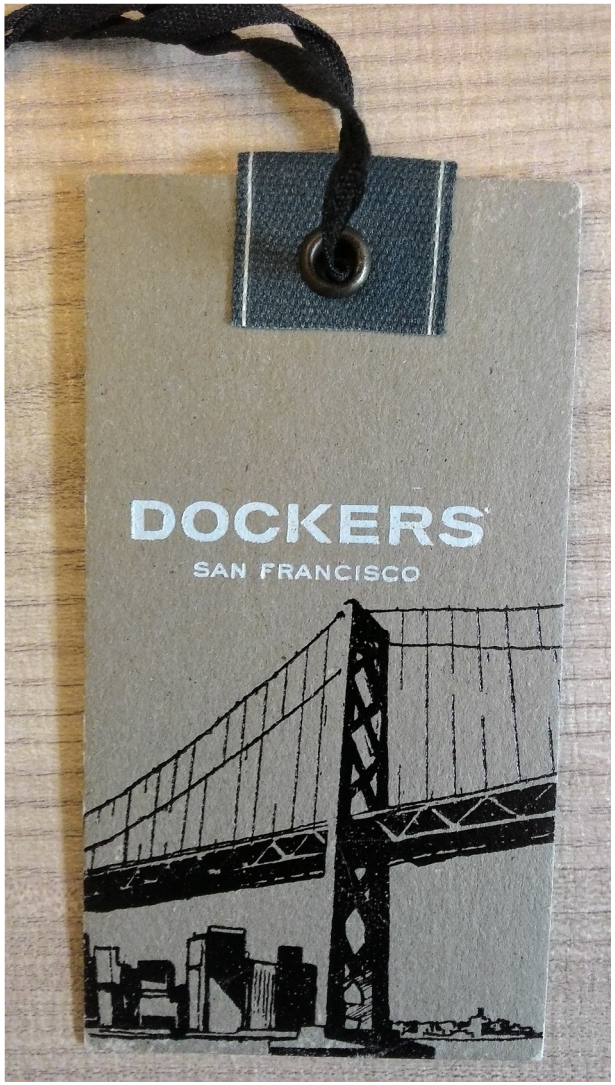
Massimiliano Dessi has more than 14 years of experience In programming,

Works in the Cloud Computing area with DevOps methods.

He's a proud father of three, Manager of GDG Sardegna,

Co-founder of JugSardegna, Author of Spring 2.5.AOP.

Dockers ?!???



Not this

Docker

```
jaiku@jaiku:~$ docker run docker/whalesay cowsay tux
 _____
< tux >
 -----
    \
     \
      \
                    ##        .
              ## ## ##       ==
           ## ## ## ##      ===
       /""""""""""""""""___/ ===
  ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
       \_______ o          __/
        \     \         __/
          \____\_______/
jaiku@jaiku:~$
```

Build, Ship, Run

An open platform for distributed applications
for developers and sysadmins

Get Started with Docker

Docker acquires Tutum
The best way to deploy and manage Dockerized apps in production
Learn more about Tutum
Try Tutum for free

Docker Webinar Series: Sign up for a Webinar
Topics include Docker technology, DevOps and customer case studies.

Announcing Docker 1.8: Read the Docker 1.8 blog post
Docker Content Trust and Toolbox Installer

Announcing DockerCon EU 2015: Register now
Join us November 16-17th in Barcelona, Spain

What is Docker?

Docker is an open platform for building, shipping and running distributed applications. It gives programmers, development teams and operations engineers the common toolbox they need to take advantage of the distributed and networked nature of modern applications.

Container idea

- A container sandboxed processes that share the same kernel as the host.
- The idea is that you can ship containers from your development environment to the deployment environment

Use Cases

- Application Development
- Test
- Packaging
- Deployment
- Application isolation
- Microservices
- Paas/Saas cloud infrastructure
- Google Cloud, Openshift, Bluemix, Amazon ECS, Cloud Foundry ...

Docker

A very lightweight wrapper around a single Unix process.

Container != Virtual Machine

- Container run at kernel level (>=3.10)

- Virtual Machine use HW with a emulation layer

- Container -> Lower overhead than VMs

Virtualization

One or more independent

machines

run virtually

on physical hardware

via an intermediation layer

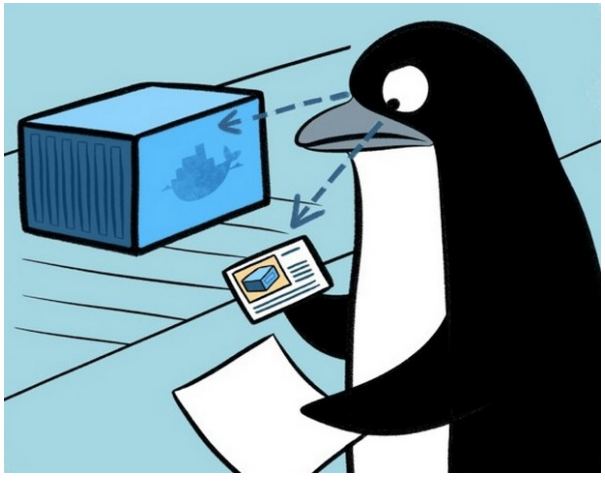
VirtualBox, VmWare, Xen

Container

containers run in user space on top of an operating system's kernel

Cgroups & Namespaces

Docker, OpenVZ, Solaris Zones, and Linux containers (lxc)



Lightweight

A container is so small because it is just a reference to a layered filesystem image and some metadata about the configuration.

Containers

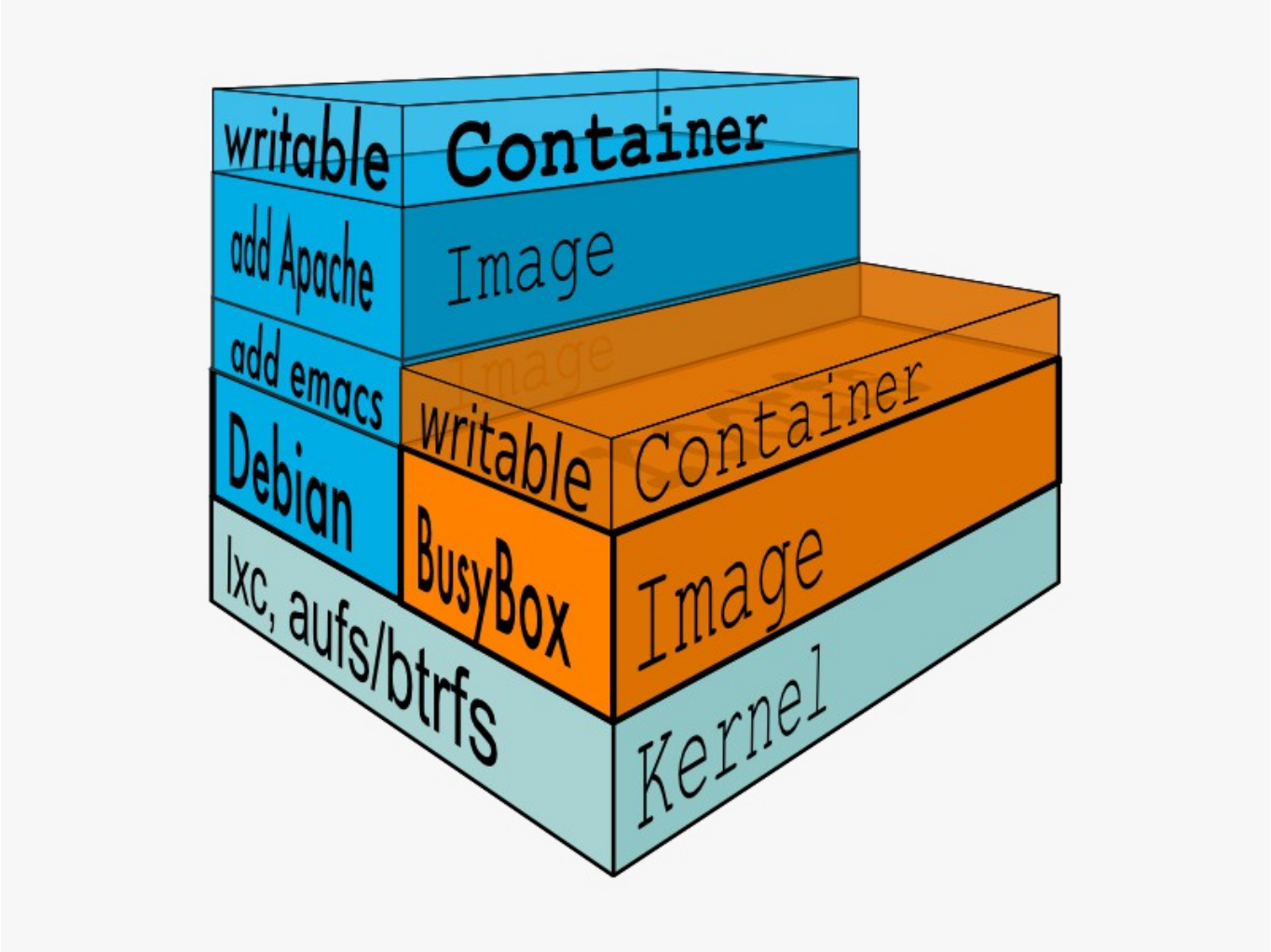
They require limited overhead and can allow a greater density of containers to run on a host.

Fast, containers start in seconds

Containers

- Copy on write model
- Layered and immutable structure (snapshot)

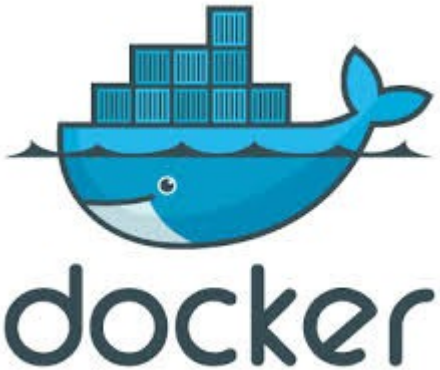
  to build one image on top of another



- A Docker image is a snapshot of a filesystem
- Repository Git-like for the images (repos)

License

- Docker is an open-source engine that automates the deployment of applications into containers released by them under the Apache 2.0 license.
- https://github.com/docker/docker/



Docker =>Golang

Basic Components

Docker client and server

Docker Images

Registries

Docker Containers

Client and Server (daemon)

```
jaiku@jaiku:~$ docker version
Client:
 Version:      1.9.1
 API version:  1.21
 Go version:   go1.4.2
 Git commit:   a34a1d5
 Built:        Fri Nov 20 13:20:08 UTC 2015
 OS/Arch:      linux/amd64

Server:
 Version:      1.9.1
 API version:  1.21
 Go version:   go1.4.2
 Git commit:   a34a1d5
 Built:        Fri Nov 20 13:20:08 UTC 2015
 OS/Arch:      linux/amd64
```

Client CLI

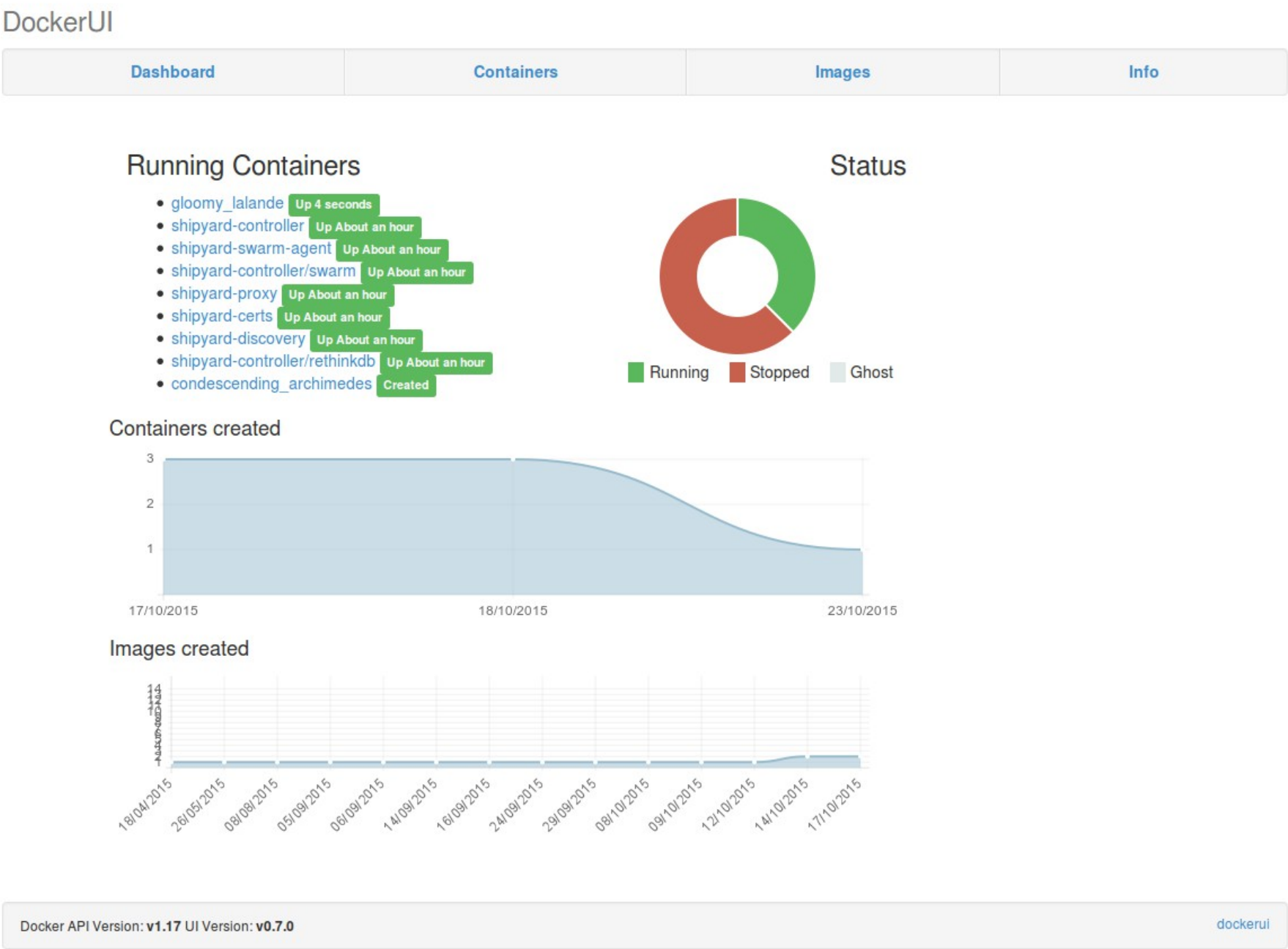
```
Commands:
    attach    Attach to a running container
    build     Build an image from a Dockerfile
    commit    Create a new image from a container's changes
    cp        Copy files/folders from a container to a HOSTDIR or to STDOUT
    create    Create a new container
    diff      Inspect changes on a container's filesystem
    events    Get real time events from the server
    exec      Run a command in a running container
    export    Export a container's filesystem as a tar archive
    history   Show the history of an image
    images    List images
    import    Import the contents from a tarball to create a filesystem image
    info      Display system-wide information
    inspect   Return low-level information on a container or image
    kill      Kill a running container
    load      Load an image from a tar archive or STDIN
    login     Register or log in to a Docker registry
    logout    Log out from a Docker registry
    logs      Fetch the logs of a container
    pause     Pause all processes within a container
    port      List port mappings or a specific mapping for the CONTAINER
    ps        List containers
    pull      Pull an image or a repository from a registry
    push      Push an image or a repository to a registry
    rename    Rename a container
    restart   Restart a running container
    rm        Remove one or more containers
    rmi       Remove one or more images
    run       Run a command in a new container
    save      Save an image(s) to a tar archive
    search    Search the Docker Hub for images
    start     Start one or more stopped containers
    stats     Display a live stream of container(s) resource usage statistics
    stop      Stop a running container
    tag       Tag an image into a repository
    top       Display the running processes of a container
    unpause   Unpause all processes within a container
    version   Show the Docker version information
    wait      Block until a container stops, then print its exit code

Run 'docker COMMAND --help' for more information on a command.
```

Server

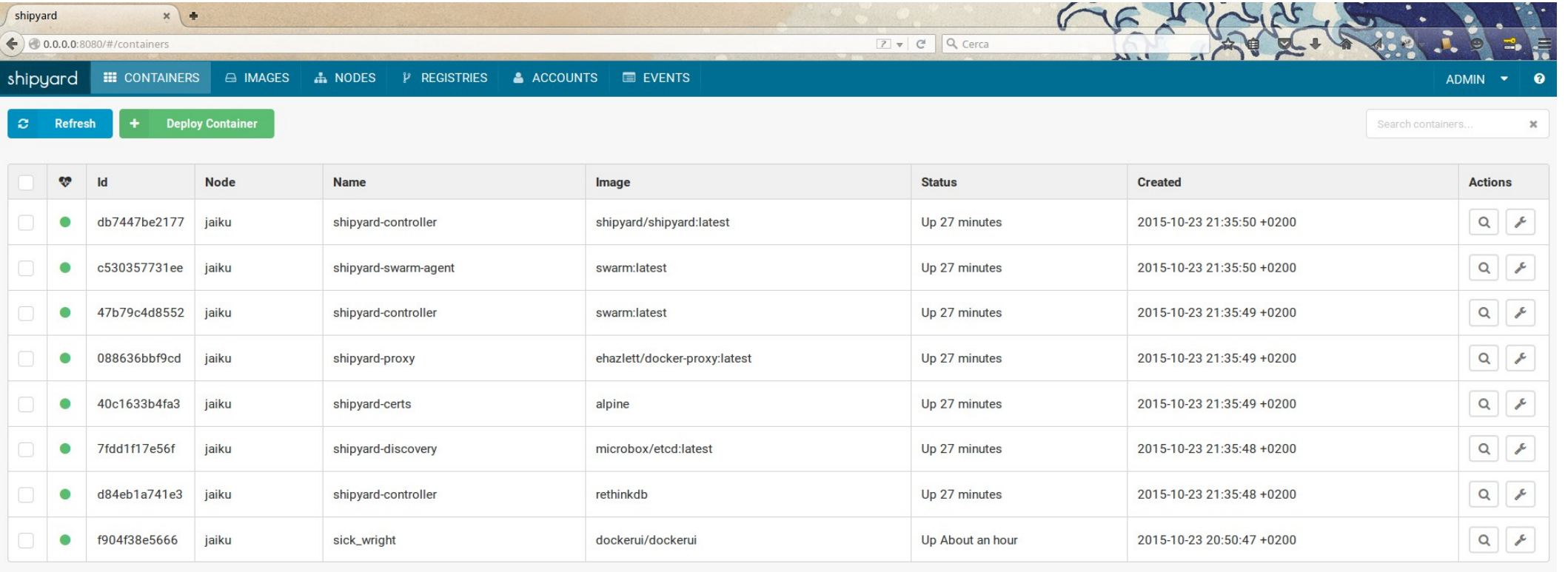
```
jaiku@jaiku:~$ docker info
Containers: 17
Images: 96
Server Version: 1.9.1
Storage Driver: aufs
 Root Dir: /var/lib/docker/aufs
 Backing Filesystem: extfs
 Dirs: 130
 Dirperm1 Supported: true
Execution Driver: native-0.2
Logging Driver: json-file
Kernel Version: 4.2.0-18-generic
Operating System: Ubuntu 15.10
CPUs: 8
Total Memory: 15.58 GiB
Name: jaiku
ID: W7YU:6XFB:IFQZ:XXOK:2FXE:X35K:JSAF:2QWV:Y4HO:G3ID:RHQN:35MU
WARNING: No swap limit support
```

Client Ui



https://github.com/crosbymichael/dockerui

Client UI



Id	Node	Name	Image	Status	Created	Actions
db7447be2177	jaiku	shipyard-controller	shipyard/shipyard:latest	Up 27 minutes	2015-10-23 21:35:50 +0200	
c530357731ee	jaiku	shipyard-swarm-agent	swarm:latest	Up 27 minutes	2015-10-23 21:35:50 +0200	
47b79c4d8552	jaiku	shipyard-controller	swarm:latest	Up 27 minutes	2015-10-23 21:35:49 +0200	
088636bbf9cd	jaiku	shipyard-proxy	ehazlett/docker-proxy:latest	Up 27 minutes	2015-10-23 21:35:49 +0200	
40c1633b4fa3	jaiku	shipyard-certs	alpine	Up 27 minutes	2015-10-23 21:35:49 +0200	
7fdd1f17e56f	jaiku	shipyard-discovery	microbox/etcd:latest	Up 27 minutes	2015-10-23 21:35:48 +0200	
d84eb1a741e3	jaiku	shipyard-controller	rethinkdb	Up 27 minutes	2015-10-23 21:35:48 +0200	
f904f38e5666	jaiku	sick_wright	dockerui/dockerui	Up About an hour	2015-10-23 20:50:47 +0200	

http://shipyard-project.com/

Client Minecraft

https://github.com/docker/dockercraft

Image

- Docker image consist of one or more filesystem layers and some metadata (Dockerfile) that represent all the files required to run a Dockerized application.

Image

- A native Linux container format (libcontainer)
- Linux kernel namespaces, => isolation for filesystems, processes, and networks.
- Filesystem isolation: each container is its own root filesystem
- Process isolation: each container runs in its own process environment
- Network isolation: separate virtual interfaces and IP addressing between containers.

Container

- A Docker container is a Linux container that has been instantiated from a Docker image.
- A container have name and a tag.
- The tag is used to identify a particular release of an image.

With src files

```
FROM desmax74/ubuntu-ansible.14.04

RUN apt-get update && apt-get install -y  g++  gcc  libc6-dev  make curl ca-certificates net-
tools
&& rm -rf /var/lib/apt/lists/* && apt-get clean

ENV GOLANG_VERSION 1.5.1
ENV GOLANG_DOWNLOAD_URL https://golang.org/dl/go$GOLANG_VERSION.src.tar.gz
ENV GOLANG_DOWNLOAD_SHA1 0df564746d105f4180c2b576a1553ebca9d9a124

RUN curl -fsSL "$GOLANG_DOWNLOAD_URL" -o golang.tar.gz \
&& echo "$GOLANG_DOWNLOAD_SHA1  golang.tar.gz" | sha1sum -c - tar -C /usr/src -xzf golang.tar.gz
&& rm golang.tar.gz  && cd /usr/src/go/src && ./make.bash --no-clean 2>&1

ENV GOPATH /go
ENV PATH $GOPATH/bin:/usr/src/go/bin:$PATH

RUN mkdir -p "$GOPATH/src" "$GOPATH/bin" && chmod -R 777 "$GOPATH"
WORKDIR $GOPATH
COPY go-wrapper /usr/local/bin/
VOLUME ["/gopath/app/","/data"]
WORKDIR /gopath/app/
```

With the binary

```
#ubuntu minimal updated the 30 of sept , ansible installed

#the entire images at the end will be 325.8 MB

FROM desmax74/ubuntu-ansible.14.04


VOLUME ["/gopath/app/","/data"]


ADD goapp/src/github.com/desmax74/app/bin /gopath/app/

ADD goapp/src/github.com/desmax74/templates /gopath/app/templates


WORKDIR /gopath/app/

EXPOSE 8080

CMD ["/gopath/app/bin"]
```

Pay Attention

- Remember that every instruction creates a new Docker image layer, so it's better combine a few logically grouped commands onto a single line. It is even possible to use the ADD instruction in combination with the RUN instruction to copy a complex script to your image and then execute that script with only two commands in the Dockerfile.

Security

By default, containers use UID 0 to launch processes but everything is running on the same kernel, many types of security vulnerabilities or simple misconfiguration can give the container’s root user unauthorized access to the host’s system resources.

Basic commands (order matter)

```
FROM desmax74/ubuntu-ansible.14.04
```

Base image to use

```
MAINTAINER  Massimiliano Dessi  @desmax74
```

Author information

```
LABEL  "release-name"="Spectre"
```

Key value to add useful info

```
USER paperinik
```

You can change the default behaviour, by default all commands runs as a root

Basic commands

```
ENV GOLANG_VERSION 1.5.1
```

The ENV instruction allows you to set shell variables that can be used during the build process

```
ADD *.go /repo
```

The ADD instruction is used to copy files from the local filesystem into your image.

```
WORKDIR $GOPATH
```

With the WORKDIR, you change the working directory in the image for the remaining build instructions

Basic commands

```
CMD ["supervisord", "-n"]
```

defines the command that launches the process that you want to run within the container

```
RUN [ "apt-get", " install", "-y", "nginx" ]
```

run command shell

```
EXPOSE 8080
```

expose a port on the container

```
VOLUME ["/gopath/app/","/data"]
```

Persistent volume

Basic commands

```
docker run --name some-mysql -e
MYSQL_ROOT_PASSWORD=password -d mysql:latest
```

```
docker run --name some-wordpress --link some-
mysql:mysql -d wordpress
```

The --link flag creates a client-service link between two containers.

The flag takes two arguments: the

container name to link and an alias for the link.

```
docker inspect <image_id>
```

detailed infos

History

```
jaiku@jaiku:/data/docker/composeapp$ docker history dockerui/dockerui
IMAGE               CREATED             CREATED BY                                      SIZE
ff2ae825a2ff        10 weeks ago        /bin/sh -c #(nop) ENTRYPOINT &{["/dockerui"]}   0 B
5e7ddffda4f3        10 weeks ago        /bin/sh -c #(nop) EXPOSE 9000/tcp               0 B
b20c31e7bb48        10 weeks ago        /bin/sh -c #(nop) COPY dir:9f287d36ea6a564760   1.155 MB
bb7350bee1e5        10 weeks ago        /bin/sh -c #(nop) COPY file:adf5aa7b9b07b65f5   4.267 MB
```

Container naming

By default, Docker randomly names your container by combining an adjective with the name of a famous person.

sad_booth furious_kare agitated_cray hungry_elion

loving_fermi compassionate_gates mad_ramanujan

cranky_kalam elegant_goldberg

Backup restore & logs

```
docker export $CONTAINER_ID > $CONTAINER_ID-
backup.tar
```

```
docker import - <name>/$CONTAINER_ID-backup <
$CONTAINER_ID-backup.tar
```

```
docker logs -f <container_id>
```

Steps

- Write a Dockerfile
- Build and tag the image

```
docker build -t desmax/<app> .
```

- Run the container

```
docker run -it desmax74/<app>
```

- Push the image on the repo

Repos


Docker Hub
https://hub.docker.com/explore/
Cerca
Explore
Help
Search
Sign up
Log In
Explore Official Repositories
centos
official
1.5 K
STARS
2.4 M
PULLS
DETAILS
busybox
official
325
STARS
40.3 M
PULLS
DETAILS
ubuntu
official
2.5 K
STARS
27.9 M
PULLS
DETAILS
scratch
official
118
STARS
224.8 K
PULLS
DETAILS
fedora
official
227
STARS
244.3 K
PULLS
DETAILS
registry
official
445
STARS
7.5 M
PULLS
DETAILS
hipache
official
46
STARS
48.5 K
PULLS
DETAILS
docker-dev
official
27
86.7 K
DETAILS

Demo

Q & A

Contacts

https://twitter.com/desmax74

http://www.slideshare.net/desmax74

https://www.linkedin.com/in/desmax74

https://github.com/desmax74/

Thanks for your attention !