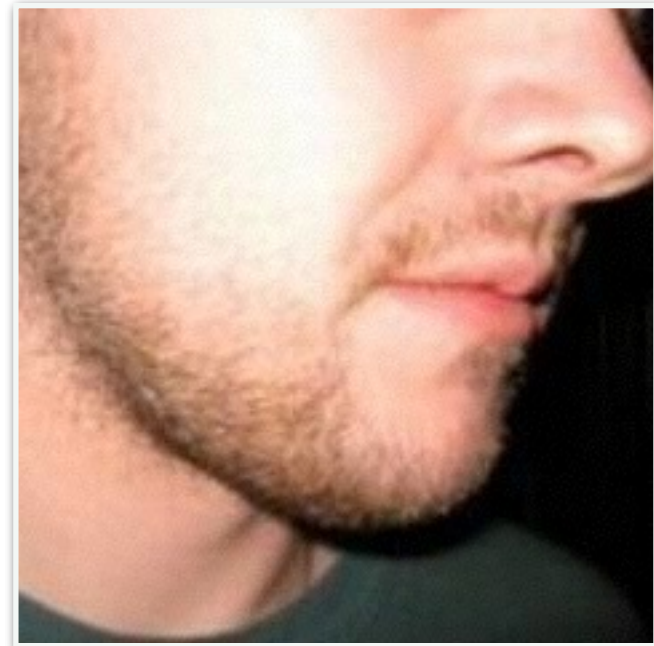


docker

Geoffrey Bachelet – @ubermuda

Who am I?

- Geoffrey Bachelet
- @ubermuda
- Then: Lagardère, SensioLabs, KnpLabs, and PMSIPilot.
- Now: Stage1, Gustave, Freelance.



Go!

"Container Engine"

Based on the Linux kernel's LXC instructions

Processes and resources isolation

"chroot" on steroids / super lightweight VMs

Containers vs VMs

LXC / Jails / etc

Use the host's kernel

Boots in seconds

0 overhead, almost

Easy to pass around

Hypervisor

Boots a complete OS

Boots in... more time.

All an OS' overhead

Several Go images

Docker vs Vagrant

Less mature (1 year)

379 contributors

Manages containers

Dockerfile (homegrown DSL)

DIY

Mature (4 years)

338 contributors

Manages... VMs.

Vagrantfile (Ruby)

Chef, Puppet, etc

```
$ echo "deb https://get.docker.io/ubuntu docker main" \  
    > /etc/apt/sources.list.d/docker.list  
  
$ apt-get update  
  
$ apt-get install lxc-docker
```

Create a container


```
$ docker search ubuntu
```

```
$ docker run -i -t stackbrew/ubuntu /bin/bash
```

```
Unable to find image 'stackbrew/ubuntu' locally
```

```
Pulling repository stackbrew/ubuntu
```

```
...
```

```
root@21d86a0b8387:/#
```

```
$ docker search ubuntu
```

```
$ docker run -i -t stackbrew/ubuntu /bin/bash
```

```
Unable to find image 'stackbrew/ubuntu' locally
```

```
Pulling repository stackbrew/ubuntu
```

```
...
```

```
root@21d86a0b8387:/#
```

```
$ docker search ubuntu
```

```
$ docker run -i -t stackbrew/ubuntu /bin/bash
```

```
Unable to find image 'stackbrew/ubuntu' locally
```

```
Pulling repository stackbrew/ubuntu
```

```
...
```

```
root@21d86a0b8387:/#
```

```
$ docker search ubuntu
```

```
$ docker run -i -t stackbrew/ubuntu /bin/bash
```

```
Unable to find image 'stackbrew/ubuntu' locally
```

```
Pulling repository stackbrew/ubuntu
```

```
...
```

```
root@21d86a0b8387:/#
```

Commit the container

```
root@21d86a0b8387:/# apt-get install nginx
```

```
root@21d86a0b8387:/# exit
```

```
$ docker ps -q -n 1
```

```
9cc9c762a0eb
```

```
$ docker commit 9cc9c762a0eb m6web/nginx
```

```
240198b750c3cc950c60005d6d24cae4fc2dbcc6c31e274574af68d4a2e8
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID
m6web/nginx	latest	240198b750c3

```
root@21d86a0b8387:/# apt-get install nginx
```

```
root@21d86a0b8387:/# exit
```

```
$ docker ps -q -n 1
```

```
9cc9c762a0eb
```

```
$ docker commit 9cc9c762a0eb m6web/nginx
```

```
240198b750c3cc950c60005d6d24cae4fc2dbcc6c31e274574af68d4a2e8
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID
m6web/nginx	latest	240198b750c3

```
root@21d86a0b8387:/# apt-get install nginx
```

```
root@21d86a0b8387:/# exit
```

```
$ docker ps -q -n 1
```

```
9cc9c762a0eb
```

```
$ docker commit 9cc9c762a0eb m6web/nginx
```

```
240198b750c3cc950c60005d6d24cae4fc2dbcc6c31e274574af68d4a2e8
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID
m6web/nginx	latest	240198b750c3


```
root@21d86a0b8387:/# apt-get install nginx
```

```
root@21d86a0b8387:/# exit
```

```
$ docker ps -q -n 1
```

```
9cc9c762a0eb
```

```
$ docker commit 9cc9c762a0eb m6web/nginx
```

```
240198b750c3cc950c60005d6d24cae4fc2dbcc6c31e274574af68d4a2e8
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID
m6web/nginx	latest	240198b750c3

```
root@21d86a0b8387:/# apt-get install nginx
```

```
root@21d86a0b8387:/# exit
```

```
$ docker ps -q -n 1
```

```
9cc9c762a0eb
```

```
$ docker commit 9cc9c762a0eb m6web/nginx
```

```
240198b750c3cc950c60005d6d24cae4fc2dbcc6c31e274574af68d4a2e8
```

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID
m6web/nginx	latest	240198b750c3

Re-use the image

```
$ docker run -p 80 m6web/nginx nginx -g 'daemon off;'
```

```
$ docker ps
```

```
CONTAINER ID    ...    PORTS
```

```
923cb190dbc3    ...    0.0.0.0:49155->80/tcp
```

```
$ curl http://localhost:49155
```

```
<html>
```

```
<head>
```

```
<title>Welcome to nginx!</title>
```

```
...
```

Dockerfile

```
FROM stackbrew/ubuntu
```

```
RUN apt-get update -y
```

```
RUN apt-get install nginx -y
```

```
RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf
```

```
EXPOSE 80
```

```
ENTRYPOINT ["nginx"]
```

FROM [stackbrew/ubuntu](#)

RUN apt-get update -y

RUN apt-get install nginx -y

RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf

EXPOSE 80

ENTRYPOINT ["nginx"]


```
FROM stackbrew/ubuntu
```

```
RUN apt-get update -y
```

```
RUN apt-get install nginx -y
```

```
RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf
```

```
EXPOSE 80
```

```
ENTRYPOINT ["nginx"]
```

```
FROM stackbrew/ubuntu
```

```
RUN apt-get update -y
```

```
RUN apt-get install nginx -y
```

```
RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf
```

```
EXPOSE 80
```

```
ENTRYPOINT ["nginx"]
```

```
FROM stackbrew/ubuntu
```

```
RUN apt-get update -y
```

```
RUN apt-get install nginx -y
```

```
RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf
```

```
EXPOSE 80
```

```
ENTRYPOINT ["nginx"]
```

```
$ docker run \  
    -p 8000:80 \  
    -v $(pwd):/usr/share/nginx/www \  
    m6web/nginx
```

```
$ echo 'Hello, world!' > index.html
```

```
$ curl http://localhost:8000
```

```
Hello, world!
```

```
$ docker run \  
    -p 8000:80 \  
    -v $(pwd):/usr/share/nginx/www \  
    m6web/nginx
```

```
$ echo 'Hello, world!' > index.html
```

```
$ curl http://localhost:8000
```

```
Hello, world!
```

```
$ docker run \  
    -p 8000:80 \  
    -v $(pwd):/usr/share/nginx/www \  
    m6web/nginx
```

```
$ echo 'Hello, world!' > index.html
```

```
$ curl http://localhost:8000
```

```
Hello, world!
```

```
$ docker run \  
    -p 8000:80 \  
    -v $(pwd):/usr/share/nginx/www \  
    m6web/nginx
```

```
$ echo 'Hello, world!' > index.html
```

```
$ curl http://localhost:8000
```

```
Hello, world!
```

Now what?


```
FROM stackbrew/ubuntu
```

```
ENV DEBIAN_FRONTEND noninteractive
```

```
RUN apt-get update -y
```

```
RUN apt-get install -y daemontools curl nginx \  
    php5-fpm php5-cli php5-mysqlnd php5-intl \  
    mysql-server
```

```
RUN curl -sS https://getcomposer.org/installer | php
```

```
RUN mv composer.phar /usr/local/bin/composer
```

```
RUN echo "daemonize=no" > /etc/php5/fpm/pool.d/daemonize.conf
```

```
RUN echo "\ndaemon off;" >> /etc/nginx/nginx.conf
```

```
ADD services /srv/services
```

```
RUN find /srv/services -name run -exec chmod +x {} \;
```

```
ADD nginx.conf /etc/nginx/sites-enabled/default
```

```
ADD php.ini /etc/php5/fpm/php.ini
```

```
ADD php.ini /etc/php5/cli/php.ini
```

```
ADD entrypoint.sh /usr/local/bin/entrypoint.sh
```

```
EXPOSE 80
```

```
ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]
```

```
FROM stackbrew/ubuntu
```

```
ENV DEBIAN_FRONTEND noninteractive
```

```
RUN apt-get update -y
```

```
RUN apt-get install -y daemontools curl nginx \
    php5-fpm php5-cli php5-mysqlnd php5-intl \
    mysql-server
```

```
RUN echo "daemonize=no" \
    > /etc/php5/fpm/pool.d/daemonize.conf
```

```
RUN echo "\ndaemon off;" \
    >> /etc/nginx/nginx.conf
```

```
FROM stackbrew/ubuntu
```

```
ENV DEBIAN_FRONTEND noninteractive
```

```
RUN apt-get update -y
```

```
RUN apt-get install -y daemontools curl nginx \
    php5-fpm php5-cli php5-mysqlnd php5-intl \
    mysql-server
```

```
RUN echo "daemonize=no" \
    > /etc/php5/fpm/pool.d/daemonize.conf
```

```
RUN echo "\ndaemon off;" \
    >> /etc/nginx/nginx.conf
```

```
RUN curl -sS https://getcomposer.org/installer | php
```

```
RUN mv composer.phar /usr/local/bin/composer
```

```
ADD services /srv/services
```

```
RUN find /srv/services -name run -exec chmod +x {} \;
```

```
=> services/mysqld/run
```

```
#!/bin/bash
```

```
exec mysqld
```

```
=> services/nginx/run
```

```
#!/bin/bash
```

```
exec nginx
```

```
=> services/php5-fpm/run
```

```
#!/bin/bash
```

```
exec php5-fpm
```

```
ADD nginx.conf /etc/nginx/sites-enabled/default
```

```
ADD php.ini /etc/php5/fpm/php.ini
```

```
ADD php.ini /etc/php5/cli/php.ini
```

```
ADD entrypoint.sh /usr/local/bin/entrypoint.sh
```

```
server {  
    listen      80;  
  
    server_name  m6web;  
    access_log   /var/log/nginx/access.log;  
    error_log    /var/log/nginx/error.log;  
  
    root /var/www/web/  
    index app_dev.php;  
  
    location / {  
        try_files $uri $uri/ /app_dev.php?$query_string;  
    }  
  
    location ~ [^/]\.php(/|$) {  
        fastcgi_pass 127.0.0.1:9000;  
        include fastcgi_params;  
    }  
}
```

```
#!/bin/bash
```

```
cd /var/www
```

```
composer install
```

```
exec svscan /srv/services
```



```
EXPOSE 80
```

```
ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]
```

```
docker run \  
    -p 8000:80 \  
    -v /project:/var/www \  
    m6web/symfony2
```

```
alias drun='docker run -t \  
    -p 8000:80 \  
    -v $(pwd):/var/www \  
    m6web/symfony2'
```

```
$ cd /project
```

```
$ drun
```

```
Loading composer repositories with package information
```

```
Installing dependencies (including require-dev) from lock file
```

```
...
```

```
$ curl http://localhost:8000
```

Container Links

```
$ docker run -name redis m6web/redis
```

```
$ docker run ... \  
    -link redis:redis \  
    m6web/symfony2
```

```
$ docker run -name redis m6web/redis
```

```
$ docker run ... \  
    -link redis:redis \  
    m6web/symfony2
```

REDIS_PORT=tcp://172.17.0.44:6379

REDIS_PORT_6379_TCP_PROTO=tcp

REDIS_PORT_6379_TCP_PORT=6379

REDIS_PORT_6379_TCP=tcp://172.17.0.44:6379

REDIS_NAME=/crimson_squirrel9/redis

REDIS_PORT_6379_TCP_ADDR=172.17.0.44

REDIS_PORT=tcp://172.17.0.44:6379

REDIS_PORT_6379_TCP_PROTO=tcp

REDIS_PORT_6379_TCP_PORT=6379

REDIS_PORT_6379_TCP=tcp://172.17.0.44:6379

REDIS_NAME=/crimson_squirrel9/redis

REDIS_PORT_6379_TCP_ADDR=172.17.0.44

app/config/config.yml

imports:

- { resource: parameters.php }

app/config/parameters.php

```
<?php
```

```
$container->setParameter(  
    'redis_host',  
    getenv( 'REDIS_PORT_6379_TCP_ADDR' )  
);
```

app/config/parameters.php

```
<?php
```

```
$container->setParameter(  
    'redis_host',  
    getenv('REDIS_PORT_6379_TCP_ADDR')  
);
```

Volume sharing

data container

FROM stackbrew/ubuntu

VOLUME ["/var/lib/mysql"]

ENTRYPOINT ["true"]

mysql container

FROM stackbrew/ubuntu

ENV DEBIAN_FRONTEND noninteractive

RUN apt-get update -y

RUN apt-get install -y mysql-server mysql-client

ADD entrypoint.sh /usr/local/bin/entrypoint.sh

ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]

```
#!/bin/bash
```

```
if [ ! -f /var/lib/mysql/ibdata1 ]; then
```

```
    mysql_install_db > /dev/null 2> /dev/null
```

```
fi
```

```
mysqld_safe > /dev/null 2> /dev/null &
```

```
while ! mysqladmin -s ping; do
```

```
    echo -n .;
```

```
    sleep 1;
```

```
done;
```

```
exec mysql
```



```
#!/bin/bash
```

```
if [ ! -f /var/lib/mysql/ibdata1 ]; then
```

```
    mysql_install_db > /dev/null 2> /dev/null
```

```
fi
```

```
mysqld_safe > /dev/null 2> /dev/null &
```

```
while ! mysqladmin -s ping; do
```

```
    echo -n .;
```

```
    sleep 1;
```

```
done;
```

```
exec mysql
```

```
#!/bin/bash
```

```
if [ ! -f /var/lib/mysql/ibdata1 ]; then
```

```
    mysql_install_db > /dev/null 2> /dev/null
```

```
fi
```

```
mysqld_safe > /dev/null 2> /dev/null &
```

```
while ! mysqladmin -s ping; do
```

```
    echo -n .;
```

```
    sleep 1;
```

```
done;
```

```
exec mysql
```

```
#!/bin/bash
```

```
if [ ! -f /var/lib/mysql/ibdata1 ]; then
```

```
    mysql_install_db > /dev/null 2> /dev/null
```

```
fi
```

```
mysqld_safe > /dev/null 2> /dev/null &
```

```
while ! mysqladmin -s ping; do
```

```
    echo -n .;
```

```
    sleep 1;
```

```
done;
```

```
exec mysql
```

```
$ docker build -t m6web/data data/
```

```
$ docker build -t m6web/mysql mysql/
```

```
$ docker run -name mysql-data m6web/data
```

```
$ docker run -i -t \
```

```
--volumes-from mysql-data \
```

```
m6web/mysql
```

And more!

- `docker run -d`, `attach`, `logs`, `top`, ...
- Docker Index / local Index
- Docker Remote API
- Dockerfile: `USER`, `WORKDIR`, `ONBUILD`, ...
- ...

Concepts recap.

- An **image** is like a VM image, it contains the hard-drive and some configuration.
- **Images** can be **pushed** and **pulled** from an **index**.
- A **container** is a running instance of an **image**.

Concepts recap.

- You can **commit** a terminated **container**, and you get a reusable **image** representing the state of that container.
- **Volumes** are like shared directories. **Containers** can share zero or many volumes.
- **Containers** can be **linked** to one another.

Thanks!

<https://github.com/ubermuda/m6web-docker/>

Geoffrey Bachelet – @ubermuda