La_plata.rb - Meetup!

Docker Development

Una respuesta a (casi) todas las preguntas sobre el desarrollo con Docker.

Lic. Juan La Battaglia

¿Qué es Docker?

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización a nivel de sistema operativo en Linux.

Snappler

Snappler

No, en serio: ¿Qué es Docker?

¿Realmente me sirve?

¿Por qué están todos tan contentos?

¿Es simple de usar?

Tip

Hace tiempo que lo observo, para **entender** qué hay que **entender**.

Hay que captar dos o tres ideas para entenderlo/usarlo.

No soy un power user.

1. Imágenes

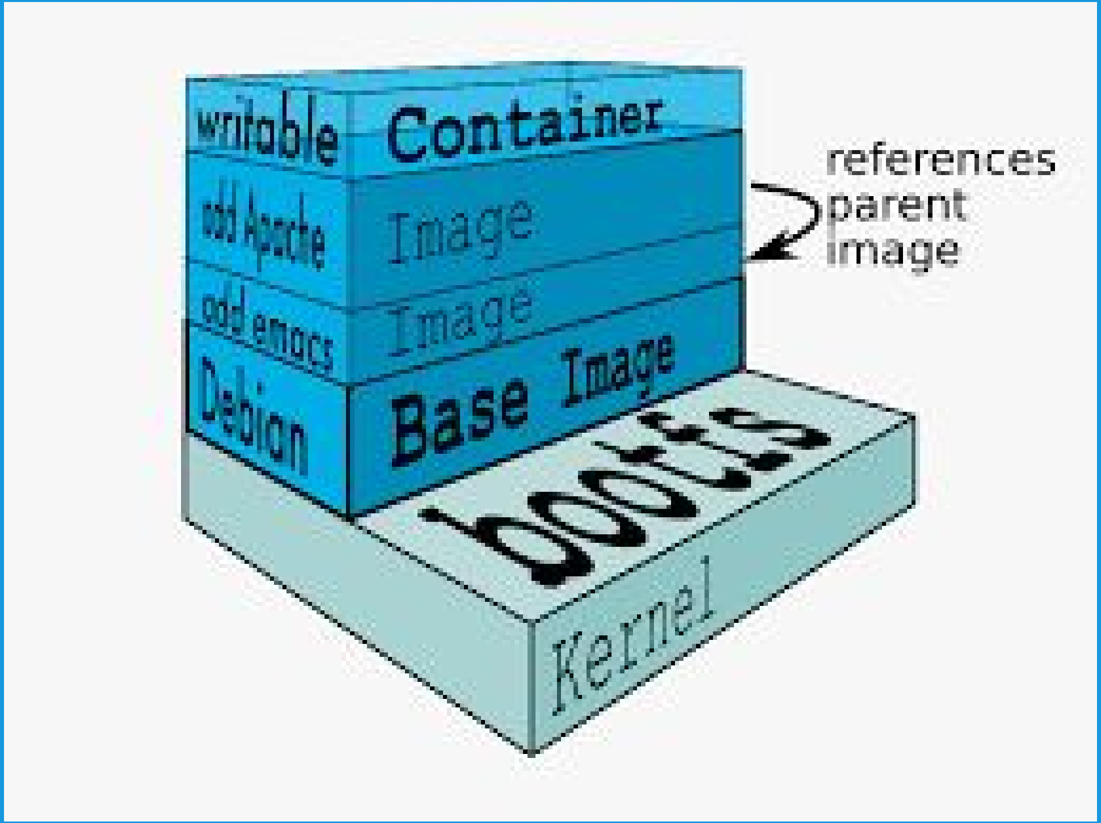
Conjunto de bits que corren. Exactamente al igual que una .iso, es un bloque de bits que puestos en el lugar "correcto", ejecutan algo.

➔ **Capas, como git.**
En vez de ser algo **monolítico**, se construyen por layers.

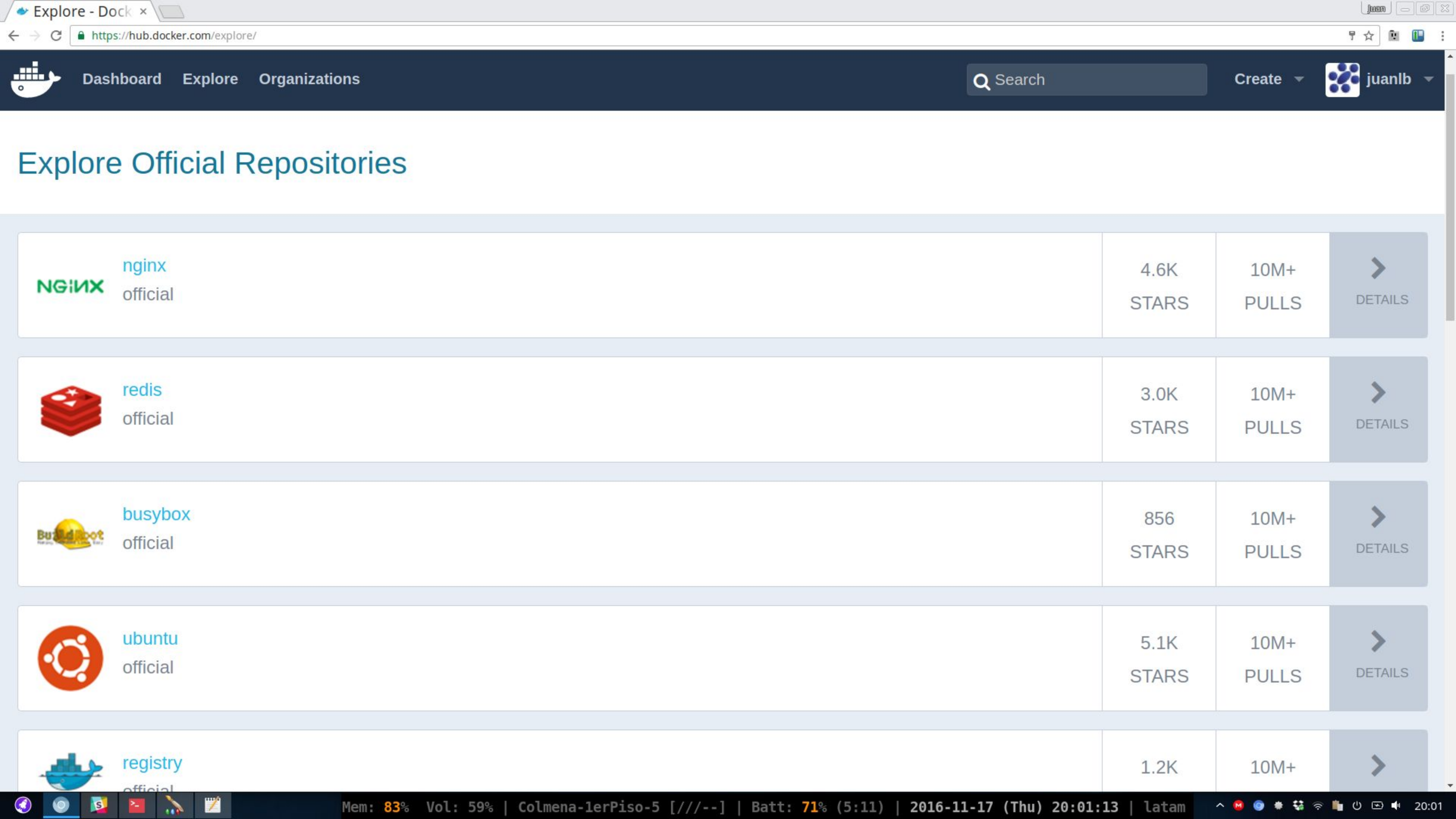
➔ **Hay un repo, como git.**
Se almacenan imágenes oficiales, seguro que andan. Registry privado.

➔ **Uno las baja y les agrega cosas.**
Adivinaron! También como git!

Snappler


writable
Container
add Apache
Image
add emacs
Image
Debian
Base Image
references parent image
bootfs
Kernel

Snappler



Explore - Dock
https://hub.docker.com/explore/
Dashboard
Explore
Organizations
Search
Create
juanlb

Explore Official Repositories

nginx official	4.6K STARS	10M+ PULLS	DETAILS
redis official	3.0K STARS	10M+ PULLS	DETAILS
busybox official	856 STARS	10M+ PULLS	DETAILS
ubuntu official	5.1K STARS	10M+ PULLS	DETAILS
registry official	1.2K	10M+	

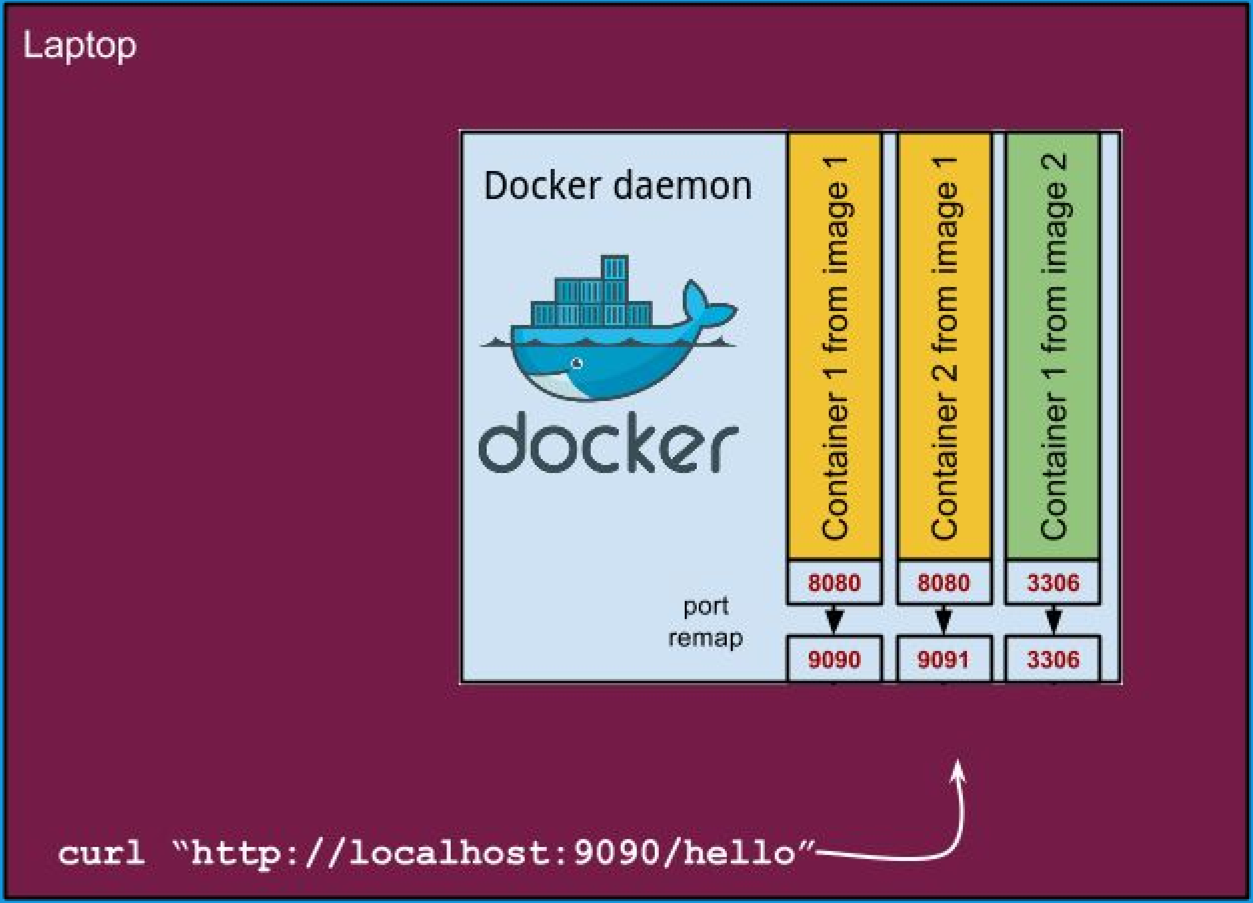
Mem: 83% Vol: 59% | Colmena-1erPiso-5 [///--] | Batt: 71% (5:11) | 2016-11-17 (Thu) 20:01:13 | latam
20:01

2. Es un servicio

Si no corre el servicio, no hay docker. Todos decían: "ejecutá **docker run zaraza**". Y yo no cachaba un fulbo.

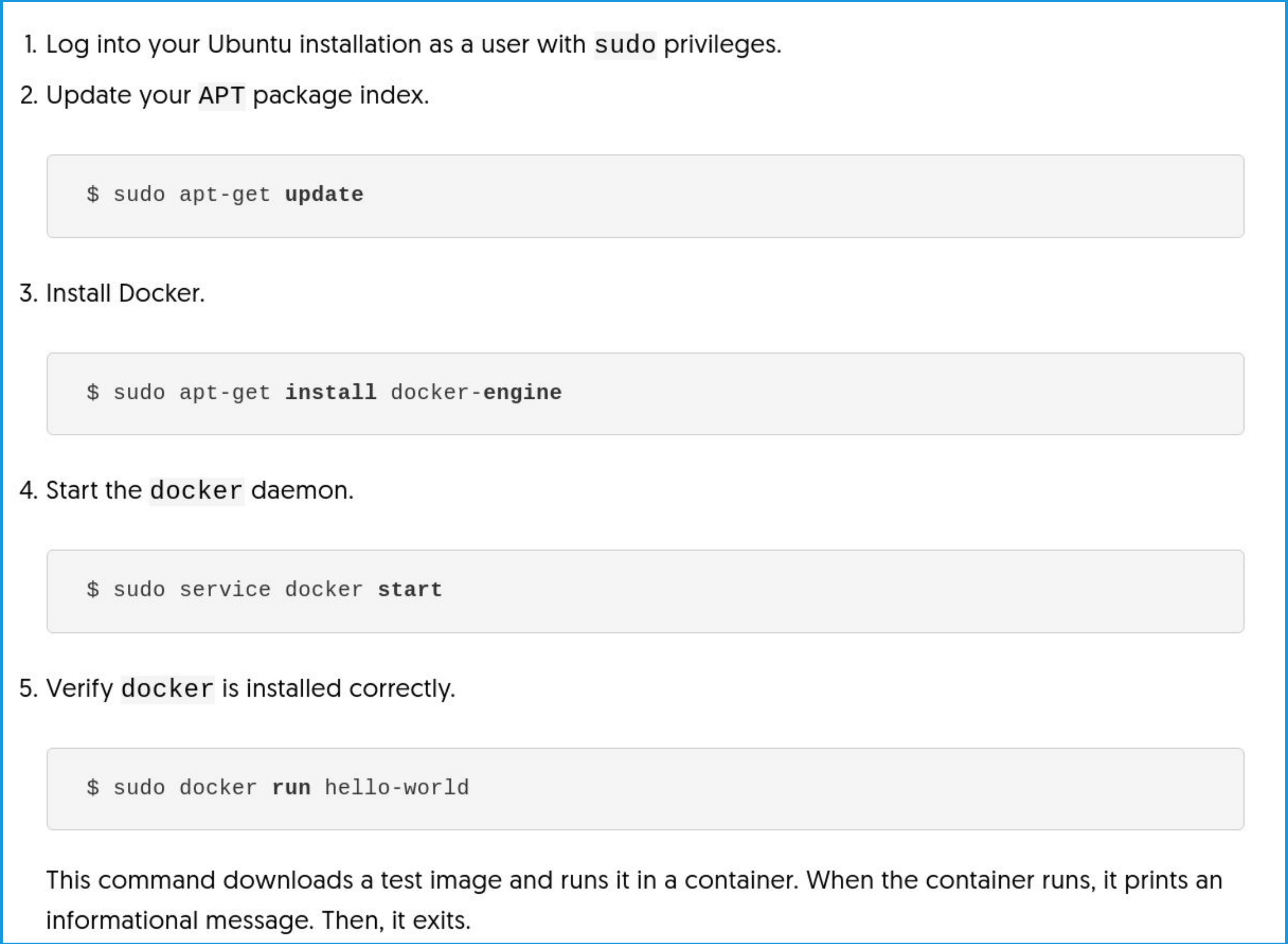
- ➔ **Instalar el servicio.**
  El chiste es que el servicio existe para todos los SO.
- ➔ **"docker run" le habla al servicio**
  Le dice que agarre una imagen (bajada y/o creada), y la haga correr. O sea, que la convierta en container.
- ➔ **Los containers son procesos.**
  Corren dentro del servicio, y "ahí adentro" se pueden hablar entre si, hay red, DNS, puertos...

Snappler



```
curl "http://localhost:9090/hello"
```

Snappler



1. Log into your Ubuntu installation as a user with `sudo` privileges.
2. Update your `APT` package index.

   ```
   $ sudo apt-get update
   ```

3. Install Docker.

   ```
   $ sudo apt-get install docker-engine
   ```

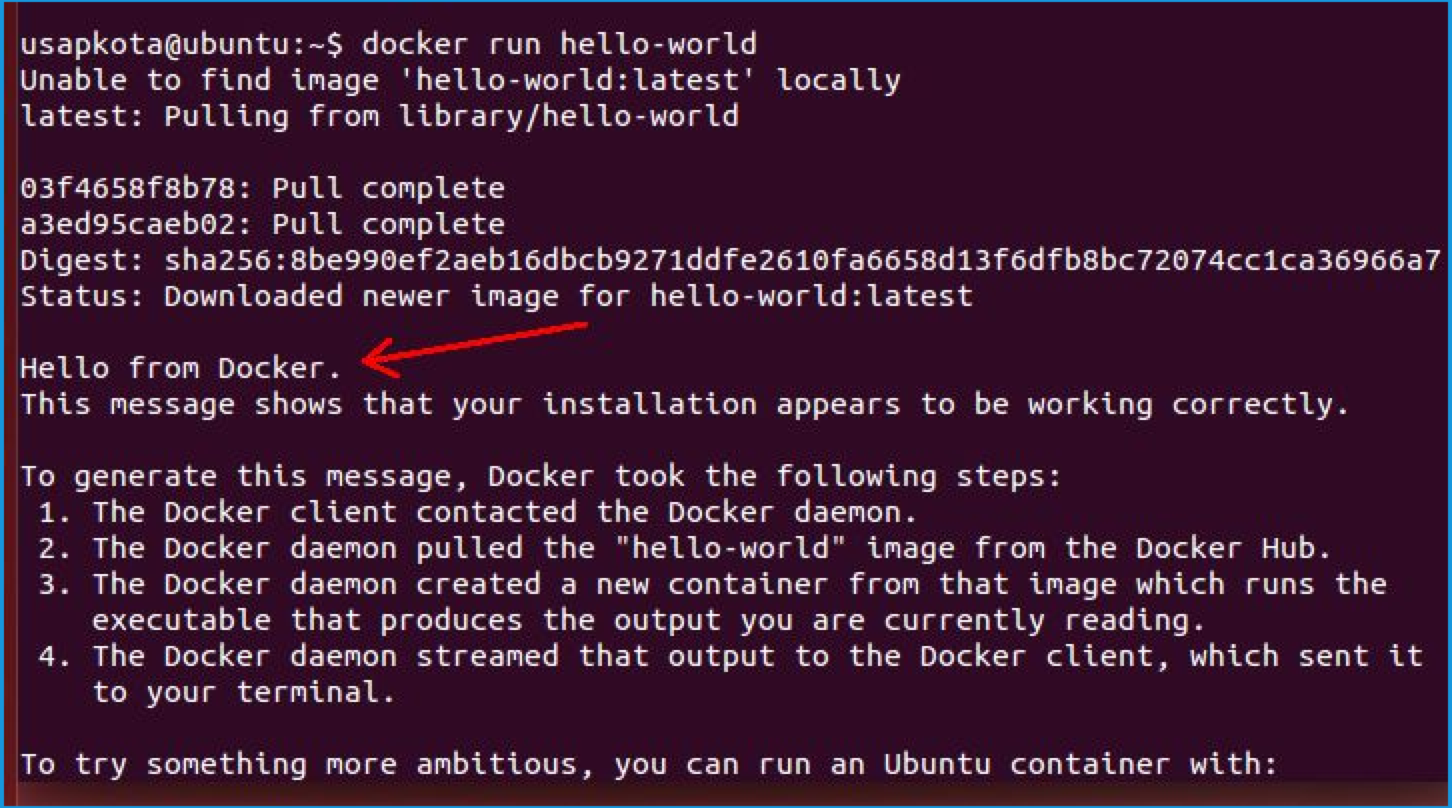
4. Start the `docker` daemon.

   ```
   $ sudo service docker start
   ```

5. Verify `docker` is installed correctly.

   ```
   $ sudo docker run hello-world
   ```

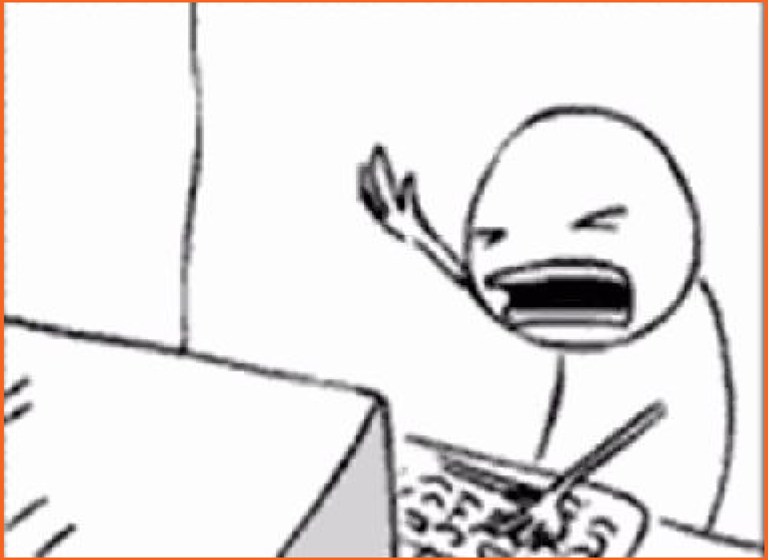
   This command downloads a test image and runs it in a container. When the container runs, it prints an informational message. Then, it exits.



```
usapkota@ubuntu:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world

03f4658f8b78: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:8be990ef2aeb16dbcb9271ddfe2610fa6658d13f6dfb8bc72074cc1ca36966a7
Status: Downloaded newer image for hello-world:latest

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.
 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:
```

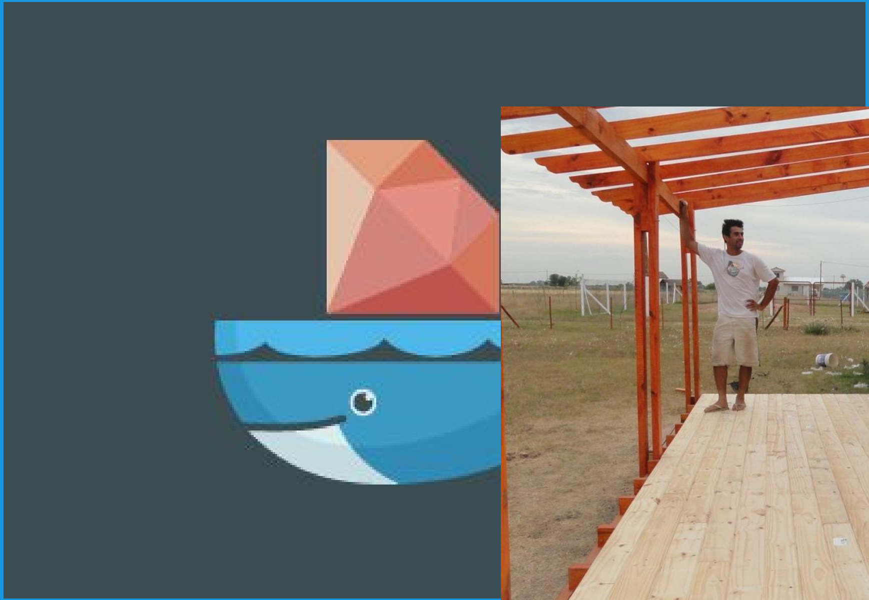
Snappler

Ahora a codear!

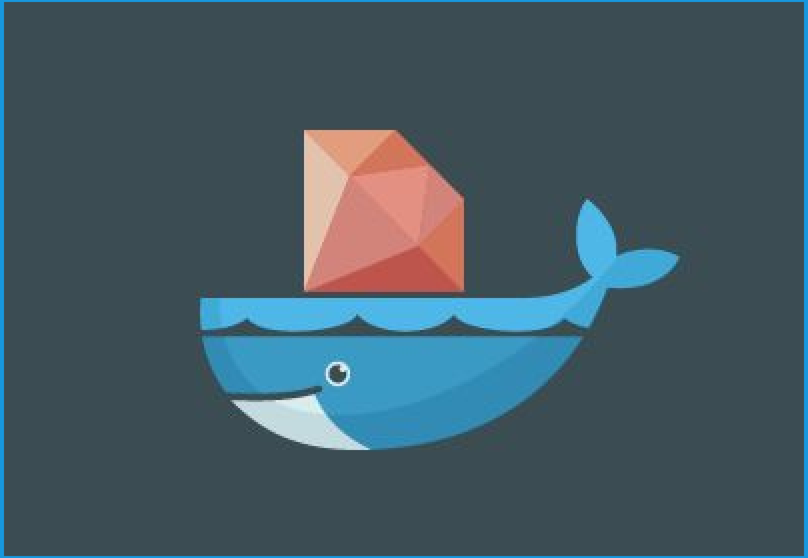


Snappler

Trayectoria



Snappler



Dockerfile

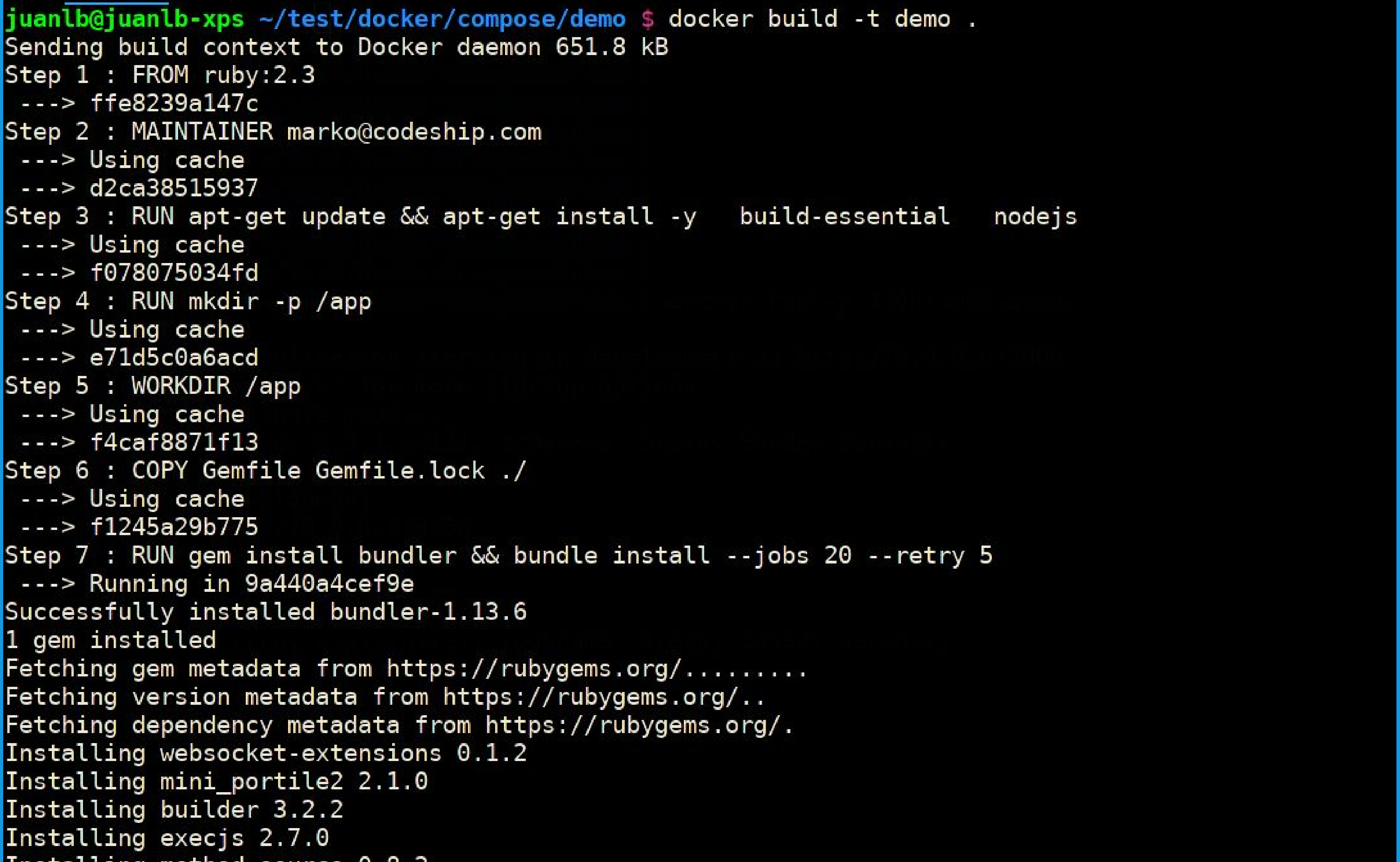
- Es una receta.
- Toma una imagen existente: FROM
- Ejecuta comando/Agrega cosas
- Cada línea = un nuevo layer
- Dice qué ejecutar al final

Si ejecuto:

- **$ docker build -t demo .**

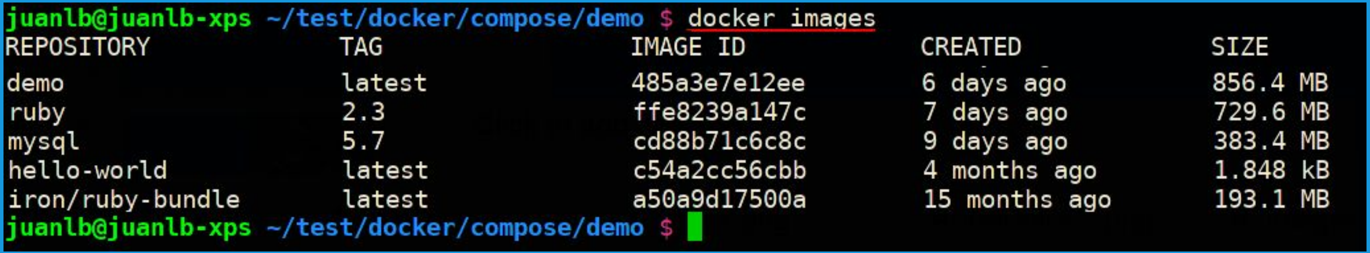
Armo la imagen con el tag "demo", y la puedo correr.

```
#./Dockerfile

FROM ruby:2.3

RUN apt-get update

RUN apt-get install -y  build-essential  nodejs


RUN mkdir -p /app

WORKDIR /app


COPY Gemfile Gemfile.lock ./

RUN gem install bundler && bundle install


COPY  .   ./

EXPOSE 3000

CMD ["bundle", "exec", "rails", "server"]
```



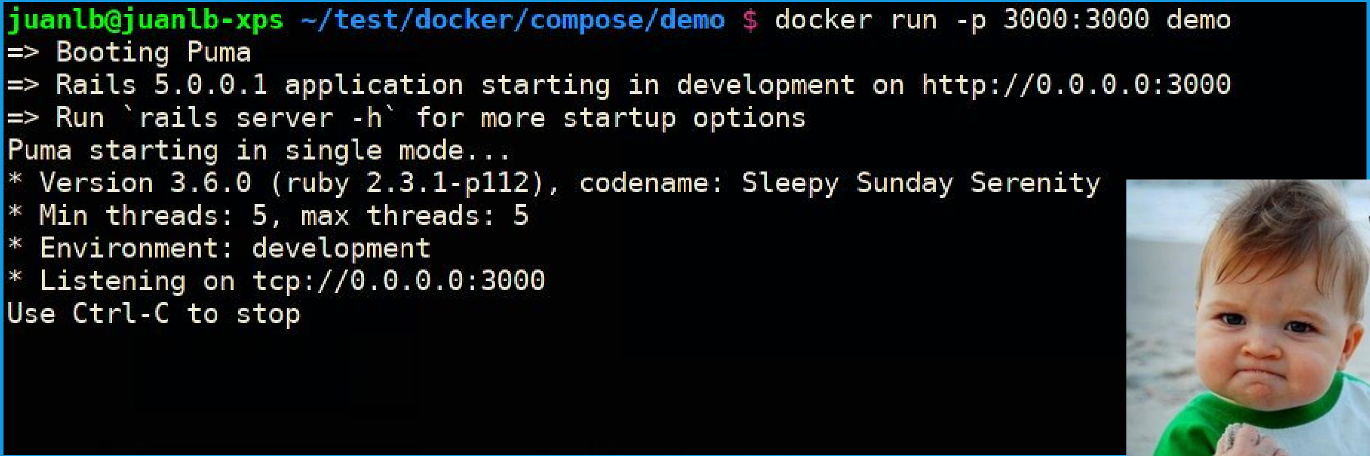
```
juanlb@juanlb-xps ~/test/docker/compose/demo $ docker build -t demo .
Sending build context to Docker daemon 651.8 kB
Step 1 : FROM ruby:2.3
 ---> ffe8239a147c
Step 2 : MAINTAINER marko@codeship.com
 ---> Using cache
 ---> d2ca38515937
Step 3 : RUN apt-get update && apt-get install -y   build-essential   nodejs
 ---> Using cache
 ---> f078075034fd
Step 4 : RUN mkdir -p /app
 ---> Using cache
 ---> e71d5c0a6acd
Step 5 : WORKDIR /app
 ---> Using cache
 ---> f4caf8871f13
Step 6 : COPY Gemfile Gemfile.lock ./
 ---> Using cache
 ---> f1245a29b775
Step 7 : RUN gem install bundler && bundle install --jobs 20 --retry 5
 ---> Running in 9a440a4cef9e
Successfully installed bundler-1.13.6
1 gem installed
Fetching gem metadata from https://rubygems.org/.........
Fetching version metadata from https://rubygems.org/..
Fetching dependency metadata from https://rubygems.org/.
Installing websocket-extensions 0.1.2
Installing mini_portile2 2.1.0
Installing builder 3.2.2
Installing execjs 2.7.0
Installing method_source 0.8.2
```

De imagen a container



```
juanlb@juanlb-xps ~/test/docker/compose/demo $ docker images
REPOSITORY            TAG              IMAGE ID         CREATED           SIZE
demo                  latest           485a3e7e12ee     6 days ago        856.4 MB
ruby                  2.3              ffe8239a147c     7 days ago        729.6 MB
mysql                 5.7              cd88b71c6c8c     9 days ago        383.4 MB
hello-world           latest           c54a2cc56cbb     4 months ago      1.848 kB
iron/ruby-bundle      latest           a50a9d17500a     15 months ago     193.1 MB
juanlb@juanlb-xps ~/test/docker/compose/demo $
```

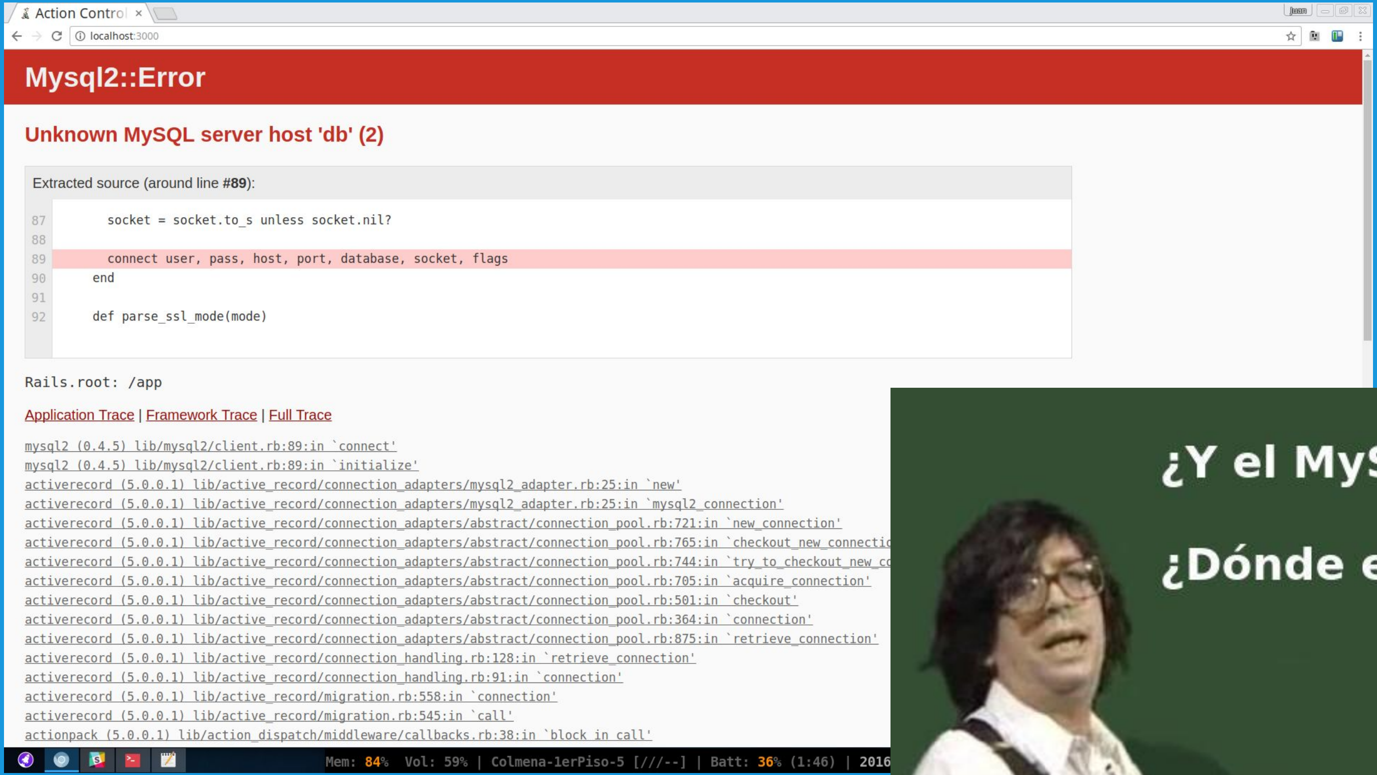
Ejecuto la imagen = container



```
juanlb@juanlb-xps ~/test/docker/compose/demo $ docker run -p 3000:3000 demo
=> Booting Puma
=> Rails 5.0.0.1 application starting in development on http://0.0.0.0:3000
=> Run `rails server -h` for more startup options
Puma starting in single mode...
* Version 3.6.0 (ruby 2.3.1-p112), codename: Sleepy Sunday Serenity
* Min threads: 5, max threads: 5
* Environment: development
* Listening on tcp://0.0.0.0:3000
Use Ctrl-C to stop
```

Snappler

Pará un cachito... y la base?


Action Control ×

localhost:3000

Mysql2::Error

Unknown MySQL server host 'db' (2)

Extracted source (around line **#89**):

```
87        socket = socket.to_s unless socket.nil?
88
89        connect user, pass, host, port, database, socket, flags
90      end
91
92      def parse_ssl_mode(mode)
```

```
Rails.root: /app
```

Application Trace | Framework Trace | Full Trace

```
mysql2 (0.4.5) lib/mysql2/client.rb:89:in `connect'
mysql2 (0.4.5) lib/mysql2/client.rb:89:in `initialize'
activerecord (5.0.0.1) lib/active_record/connection_adapters/mysql2_adapter.rb:25:in `new'
activerecord (5.0.0.1) lib/active_record/connection_adapters/mysql2_adapter.rb:25:in `mysql2_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:721:in `new_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:765:in `checkout_new_connectic
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:744:in `try_to_checkout_new_co
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:705:in `acquire_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:501:in `checkout'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:364:in `connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:875:in `retrieve_connection'
activerecord (5.0.0.1) lib/active_record/connection_handling.rb:128:in `retrieve_connection'
activerecord (5.0.0.1) lib/active_record/connection_handling.rb:91:in `connection'
activerecord (5.0.0.1) lib/active_record/migration.rb:558:in `connection'
activerecord (5.0.0.1) lib/active_record/migration.rb:545:in `call'
actionpack (5.0.0.1) lib/action_dispatch/middleware/callbacks.rb:38:in `block in call'
```

```
Mem: 84% Vol: 59% | Colmena-1erPiso-5 [///--] | Batt: 36% (1:46) | 2016
```

Snappler

3. docker-compose

Correr varios docker juntos. Es una nueva capa, que utiliza el daemon, las imágenes, los containers y les permite **relacionarse**.

- ➔ **Receta para varios dockers.**
  También tiene un archivo con instrucciones sobre qué ejecutar.
- ➔ **Identifica los containers.**
  Asigna nombre a los containers, y usa el DNS interno del docker dameon, o sea que los containers se “ven”.
- ➔ **Se levanta todo junto.**
  Con una sola instrucción, se levantan todos los containers.

Snappler

4. Volúmenes

Los datos en los containers son volátiles.
Los archivos que se escriben en un container durante su ejecución se borran al frenarlo.

➔ **Volumenes en docker.**
Se puede indicar que un directorio dentro del container está “linkeado” con un directorio real de la máquina host.

➔ **La doble Nelson.**
- Sirve para no perder los datos de una base de datos.

- Sirve para codear en vivo!

Snappler

docker-compose.yml

- También es una receta.
- Toma imagenes existentes
- Setea propiedades
- Ejecuta lo que dice la imagen
- Relaciona containers

Si ejecuto:

- **$ docker-compose build (o “up”)**

Construyo las imágenes y las puedo correr.

```
#./docker-compose.yml
app:
  build: .
  volumes:
   - . : /app
  ports:
   - "3000:3000"
  links:
   - db
  depends_on:
   - db
db:
  image: mysql:5.7
  Volumes:
   - /var/mysql/NAME  :  /var/lib/mysql
  environment:
   - MYSQL_ROOT_PASSWORD = root
  ports:
   - "4408:3306"  <- yo tengo mysql local
```

Snappler

Único cambio en la app!

DNS Interno

Los containers pueden hacerse referencia entre sí, utilizando el nombre del container del **docker-compose.yml**

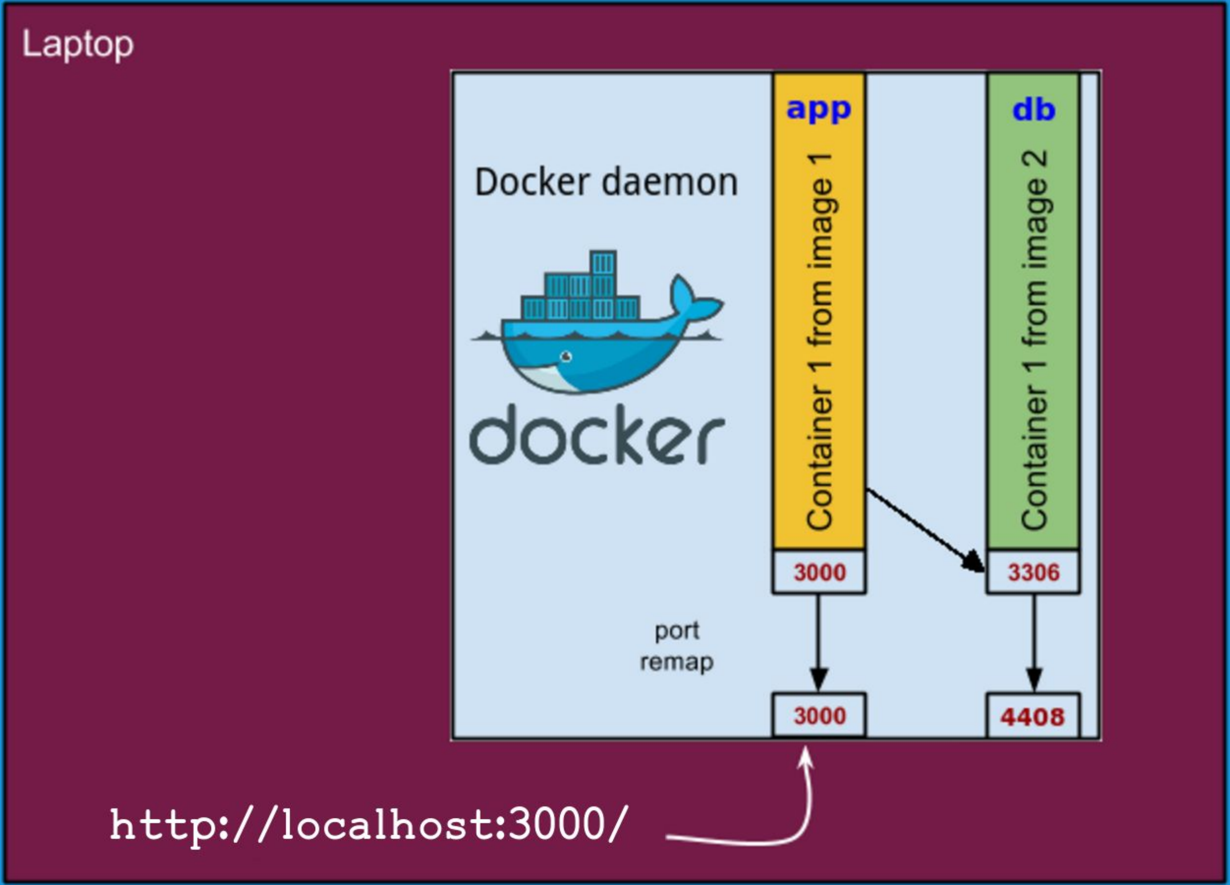
```
#./database.yml

default: &default
  adapter: mysql2
  encoding: utf8
  pool: 5
  username: root
  password: root
  # socket: /var/run/mysqld/mysqld.sock
  host: db

development:
  <<: *default
  database: demo_development
```

Snappler

Comandos de docker-compose

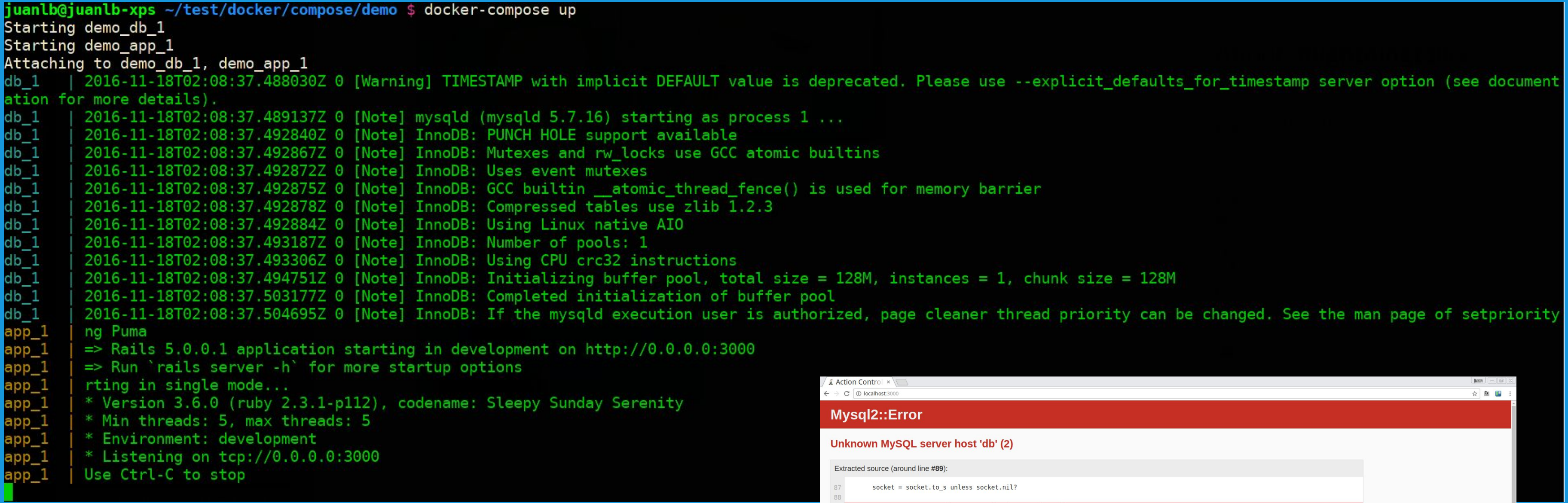
- $ docker-compose **build**
    - La primera vez (y ni siquiera….) y cuando agrego una gema
- $ docker-compose **up**
    - Levanta todos los containers y los deja corriendo
- $ docker-compose **stop/start**
    - Arranca y frena los containers
- $ docker-compose **down**
    - Borra todo

Stop o Down, eh? -> down mata el **estado** del container.

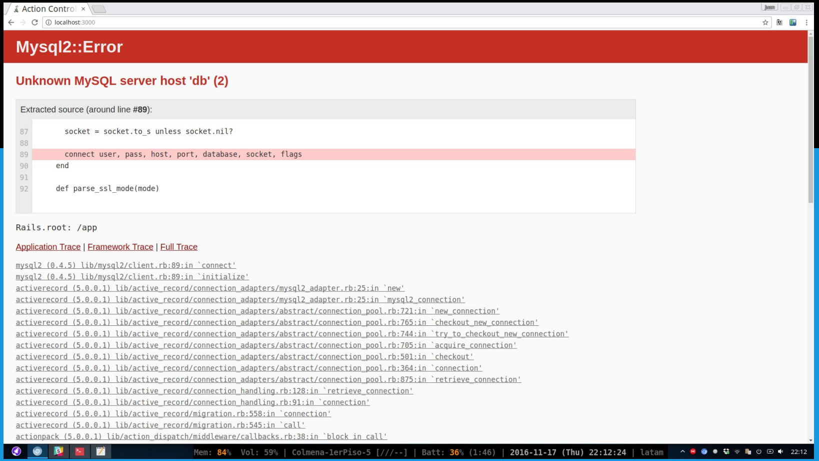

Laptop
Docker daemon
docker
app
Container 1 from image 1
db
Container 1 from image 2
3000
3306
port
remap
3000
4408
http://localhost:3000/

Snappler

Ahora si levantamos todo



```
juanlb@juanlb-xps ~/test/docker/compose/demo $ docker-compose up
Starting demo_db_1
Starting demo_app_1
Attaching to demo_db_1, demo_app_1
db_1    | 2016-11-18T02:08:37.488030Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit_defaults_for_timestamp server option (see document
ation for more details).
db_1    | 2016-11-18T02:08:37.489137Z 0 [Note] mysqld (mysqld 5.7.16) starting as process 1 ...
db_1    | 2016-11-18T02:08:37.492840Z 0 [Note] InnoDB: PUNCH HOLE support available
db_1    | 2016-11-18T02:08:37.492867Z 0 [Note] InnoDB: Mutexes and rw_locks use GCC atomic builtins
db_1    | 2016-11-18T02:08:37.492872Z 0 [Note] InnoDB: Uses event mutexes
db_1    | 2016-11-18T02:08:37.492875Z 0 [Note] InnoDB: GCC builtin __atomic_thread_fence() is used for memory barrier
db_1    | 2016-11-18T02:08:37.492878Z 0 [Note] InnoDB: Compressed tables use zlib 1.2.3
db_1    | 2016-11-18T02:08:37.492884Z 0 [Note] InnoDB: Using Linux native AIO
db_1    | 2016-11-18T02:08:37.493187Z 0 [Note] InnoDB: Number of pools: 1
db_1    | 2016-11-18T02:08:37.493306Z 0 [Note] InnoDB: Using CPU crc32 instructions
db_1    | 2016-11-18T02:08:37.494751Z 0 [Note] InnoDB: Initializing buffer pool, total size = 128M, instances = 1, chunk size = 128M
db_1    | 2016-11-18T02:08:37.503177Z 0 [Note] InnoDB: Completed initialization of buffer pool
db_1    | 2016-11-18T02:08:37.504695Z 0 [Note] InnoDB: If the mysqld execution user is authorized, page cleaner thread priority can be changed. See the man page of setpriority
app_1   | ng Puma
app_1   | => Rails 5.0.0.1 application starting in development on http://0.0.0.0:3000
app_1   | => Run `rails server -h` for more startup options
app_1   | rting in single mode...
app_1   | * Version 3.6.0 (ruby 2.3.1-p112), codename: Sleepy Sunday Serenity
app_1   | * Min threads: 5, max threads: 5
app_1   | * Environment: development
app_1   | * Listening on tcp://0.0.0.0:3000
app_1   | Use Ctrl-C to stop
```



Mysql2::Error

Unknown MySQL server host 'db' (2)

Extracted source (around line **#89**):

```
      socket = socket.to_s unless socket.nil?

      connect user, pass, host, port, database, socket, flags
    end

    def parse_ssl_mode(mode)
```

`Rails.root: /app`

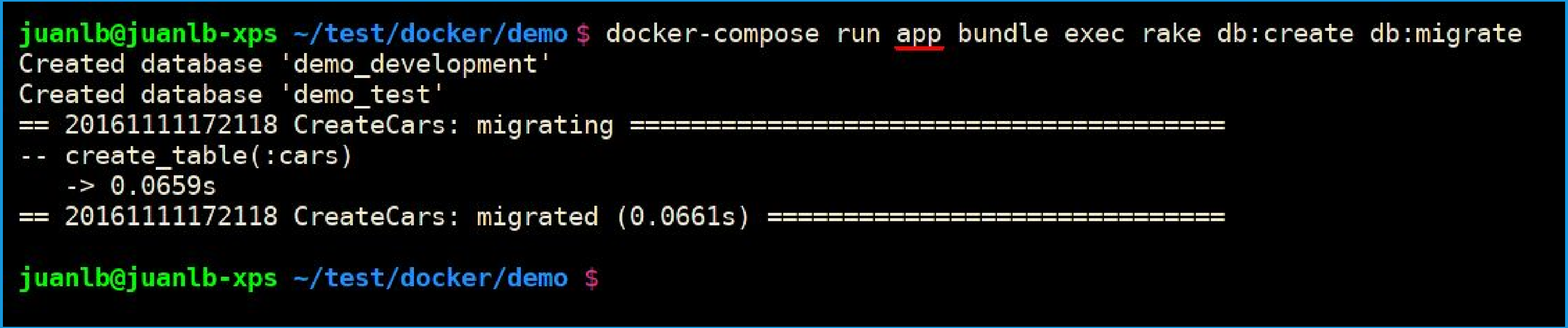
Application Trace | Framework Trace | Full Trace

```
mysql2 (0.4.5) lib/mysql2/client.rb:89:in `connect'
mysql2 (0.4.5) lib/mysql2/client.rb:89:in `initialize'
activerecord (5.0.0.1) lib/active_record/connection_adapters/mysql2_adapter.rb:25:in `new'
activerecord (5.0.0.1) lib/active_record/connection_adapters/mysql2_adapter.rb:25:in `mysql2_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:721:in `new_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:765:in `checkout_new_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:744:in `try_to_checkout_new_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:705:in `acquire_connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:501:in `checkout'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:364:in `connection'
activerecord (5.0.0.1) lib/active_record/connection_adapters/abstract/connection_pool.rb:875:in `retrieve_connection'
activerecord (5.0.0.1) lib/active_record/connection_handling.rb:128:in `retrieve_connection'
activerecord (5.0.0.1) lib/active_record/connection_handling.rb:91:in `connection'
activerecord (5.0.0.1) lib/active_record/migration.rb:558:in `connection'
activerecord (5.0.0.1) lib/active_record/migration.rb:545:in `call'
actionpack (5.0.0.1) lib/action_dispatch/middleware/callbacks.rb:38:in `block in call'
```

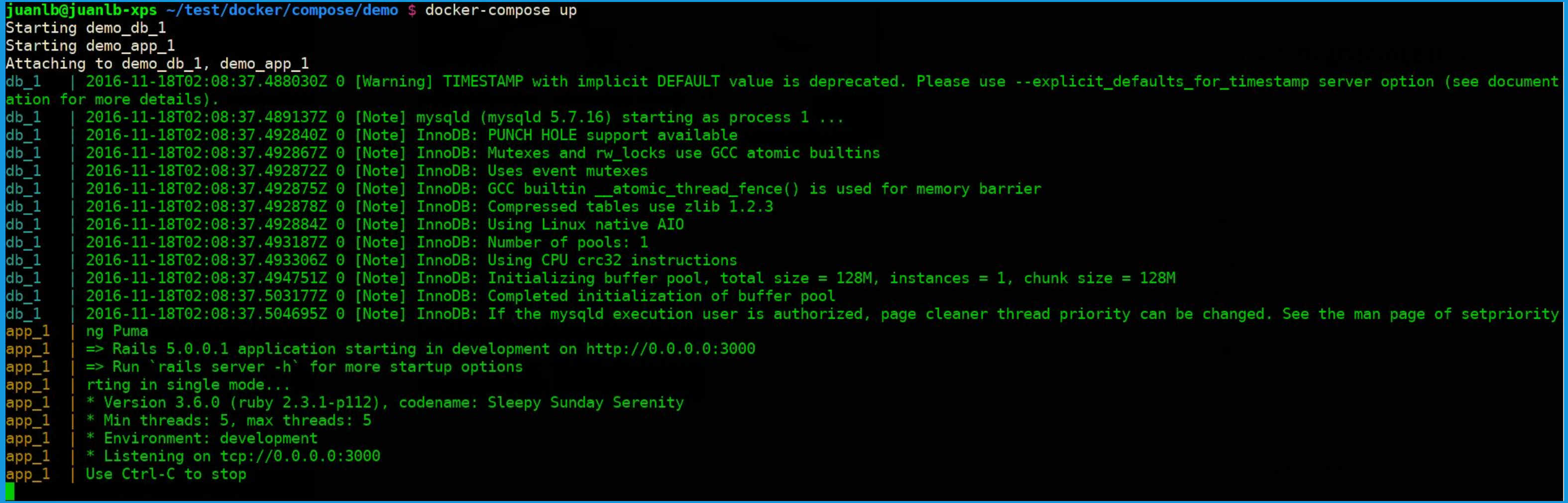
```
Mem: 84% Vol: 59% | Colmena-1erPiso-5 [///--] | Batt: 36% (1:46) | 2016-11-17 (Thu) 22:12:24 | latam     22:12
```

Snappler

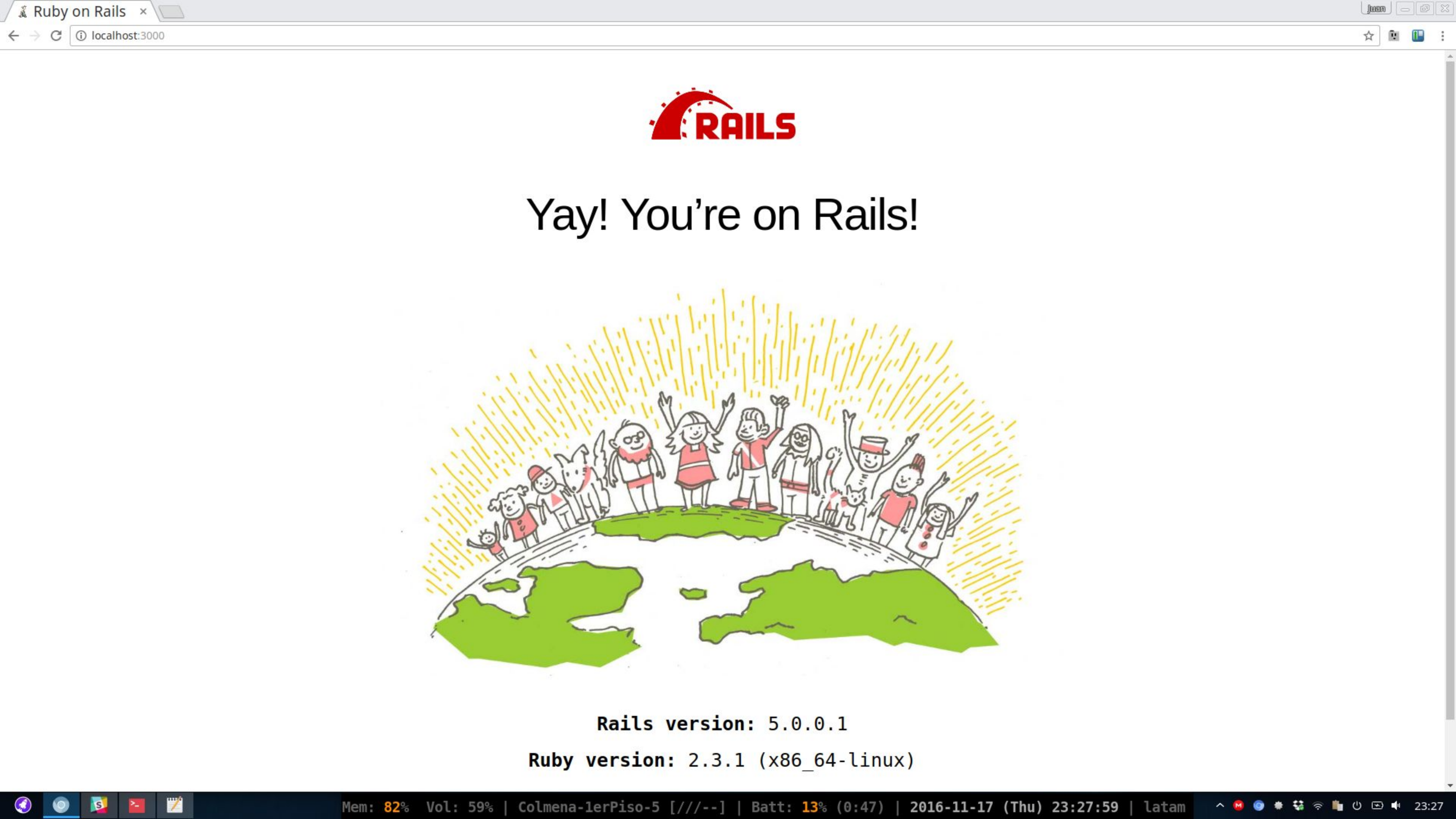
Ejecutar comandos en el container



```
juanlb@juanlb-xps ~/test/docker/demo $ docker-compose run app bundle exec rake db:create db:migrate
Created database 'demo_development'
Created database 'demo_test'
== 20161111172118 CreateCars: migrating =======================================
-- create_table(:cars)
   -> 0.0659s
== 20161111172118 CreateCars: migrated (0.0661s) ==============================

juanlb@juanlb-xps ~/test/docker/demo $
```



```
juanlb@juanlb-xps ~/test/docker/compose/demo $ docker-compose up
Starting demo_db_1
Starting demo_app_1
Attaching to demo_db_1, demo_app_1
db_1   | 2016-11-18T02:08:37.488030Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated. Please use --explicit_defaults_for_timestamp server option (see document
ation for more details).
db_1   | 2016-11-18T02:08:37.489137Z 0 [Note] mysqld (mysqld 5.7.16) starting as process 1 ...
db_1   | 2016-11-18T02:08:37.492840Z 0 [Note] InnoDB: PUNCH HOLE support available
db_1   | 2016-11-18T02:08:37.492867Z 0 [Note] InnoDB: Mutexes and rw_locks use GCC atomic builtins
db_1   | 2016-11-18T02:08:37.492872Z 0 [Note] InnoDB: Uses event mutexes
db_1   | 2016-11-18T02:08:37.492875Z 0 [Note] InnoDB: GCC builtin __atomic_thread_fence() is used for memory barrier
db_1   | 2016-11-18T02:08:37.492878Z 0 [Note] InnoDB: Compressed tables use zlib 1.2.3
db_1   | 2016-11-18T02:08:37.492884Z 0 [Note] InnoDB: Using Linux native AIO
db_1   | 2016-11-18T02:08:37.493187Z 0 [Note] InnoDB: Number of pools: 1
db_1   | 2016-11-18T02:08:37.493306Z 0 [Note] InnoDB: Using CPU crc32 instructions
db_1   | 2016-11-18T02:08:37.494751Z 0 [Note] InnoDB: Initializing buffer pool, total size = 128M, instances = 1, chunk size = 128M
db_1   | 2016-11-18T02:08:37.503177Z 0 [Note] InnoDB: Completed initialization of buffer pool
db_1   | 2016-11-18T02:08:37.504695Z 0 [Note] InnoDB: If the mysqld execution user is authorized, page cleaner thread priority can be changed. See the man page of setpriority
app_1  | ng Puma
app_1  | => Rails 5.0.0.1 application starting in development on http://0.0.0.0:3000
app_1  | => Run `rails server -h` for more startup options
app_1  | rting in single mode...
app_1  | * Version 3.6.0 (ruby 2.3.1-p112), codename: Sleepy Sunday Serenity
app_1  | * Min threads: 5, max threads: 5
app_1  | * Environment: development
app_1  | * Listening on tcp://0.0.0.0:3000
app_1  | Use Ctrl-C to stop
```



Yay! You’re on Rails!

Rails version: 5.0.0.1

Ruby version: 2.3.1 (x86_64-linux)

```
Mem: 82%  Vol: 59% | Colmena-1erPiso-5 [///--] | Batt: 13% (0:47) | 2016-11-17 (Thu) 23:27:59 | latam
```

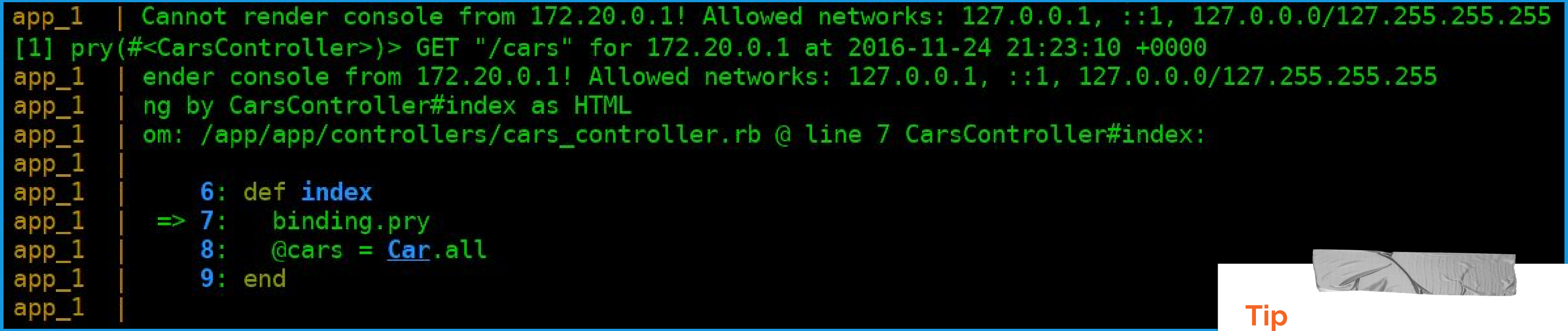
Snappler

Un momento...

¿Cómo uso **binding.pry** o **byebug** con esta ~~garcha~~ nueva tecnología?

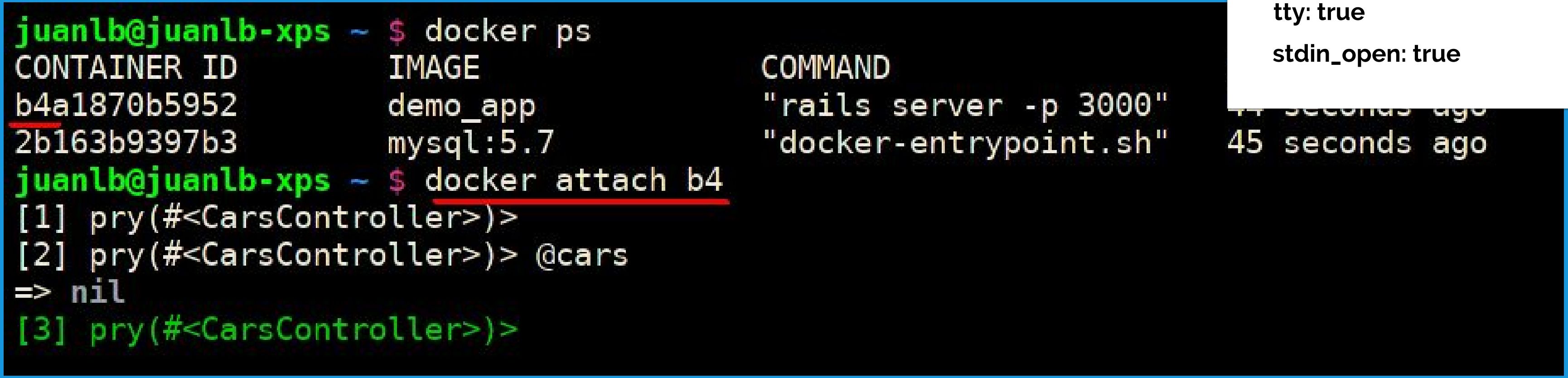
Snappler

docker attach



```
app_1  | Cannot render console from 172.20.0.1! Allowed networks: 127.0.0.1, ::1, 127.0.0.0/127.255.255.255
[1] pry(#<CarsController>)> GET "/cars" for 172.20.0.1 at 2016-11-24 21:23:10 +0000
app_1  | ender console from 172.20.0.1! Allowed networks: 127.0.0.1, ::1, 127.0.0.0/127.255.255.255
app_1  | ng by CarsController#index as HTML
app_1  | om: /app/app/controllers/cars_controller.rb @ line 7 CarsController#index:
app_1  |
app_1  |     6: def index
app_1  |  => 7:   binding.pry
app_1  |     8:   @cars = Car.all
app_1  |     9: end
app_1  |
```

En otra consola:



```
juanlb@juanlb-xps ~ $ docker ps
CONTAINER ID        IMAGE               COMMAND
b4a1870b5952        demo_app            "rails server -p 3000"   44 seconds ago
2b163b9397b3        mysql:5.7           "docker-entrypoint.sh"   45 seconds ago
juanlb@juanlb-xps ~ $ docker attach b4
[1] pry(#<CarsController>)>
[2] pry(#<CarsController>)> @cars
=> nil
[3] pry(#<CarsController>)>
```

Tip

En **docker-compose.yml** van estos parámetros:

tty: true

stdin_open: true

¿Qué ganamos?

- Nunca más problemas de incompatibilidad
- Ubuntu / Mac / luqui
- Las imágenes son oficiales y mantenidas
- Crece el índice de cancheritud y aumenta en un 17% las chances de ganar minitas/flaquitos.

Incomprobado.

Snappler

¿Qué falta?

- Agregar una gema y no tener que instalar todas de nuevo.
- Conciencia de que es algo nuevo.

Pero nada que impida empezar hoy!

Snappler

Probemos algo nuevo antes de navidad...

Snappler