



# Despliegue de Aplicaciones: Taller de Docker (1)

Evolución y Gestión de la Configuración

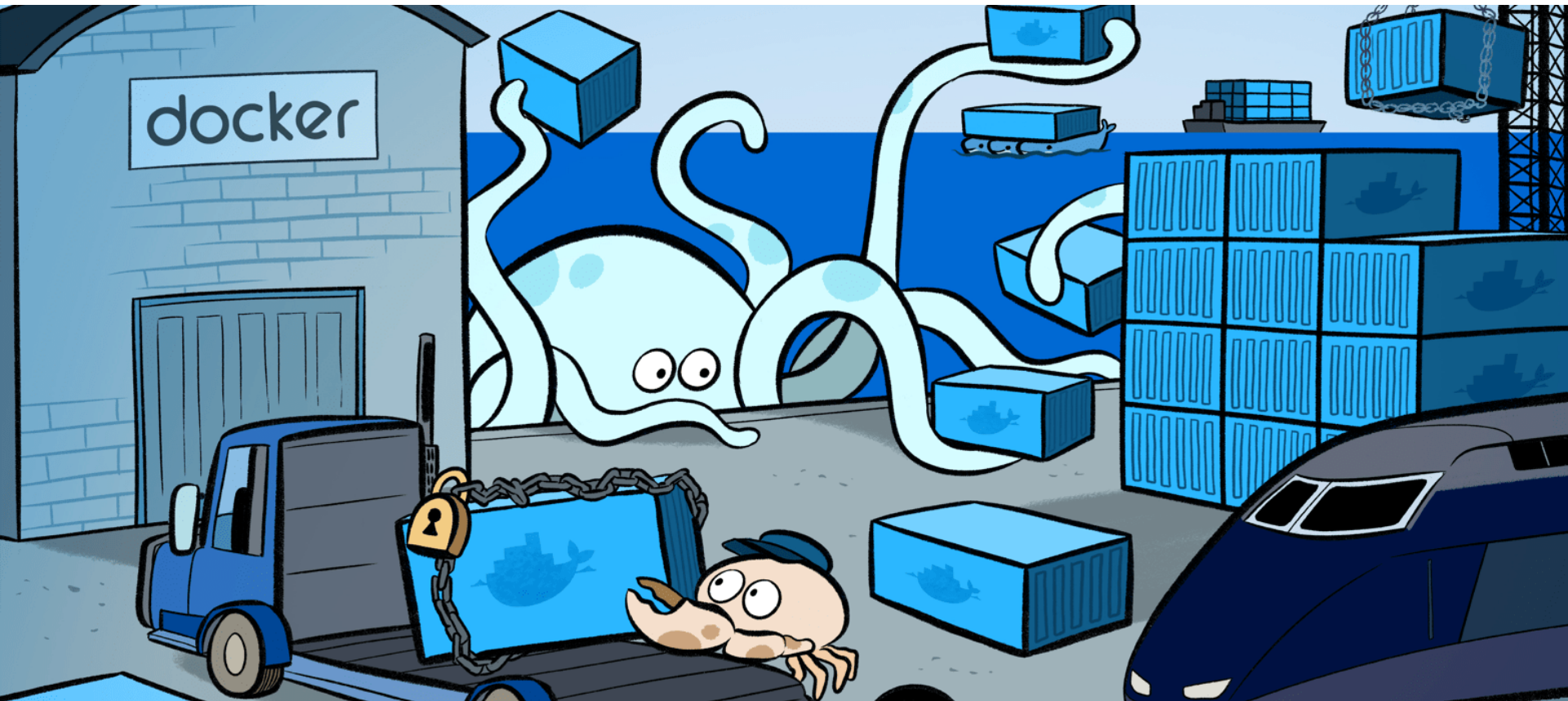
## Descarga Docker

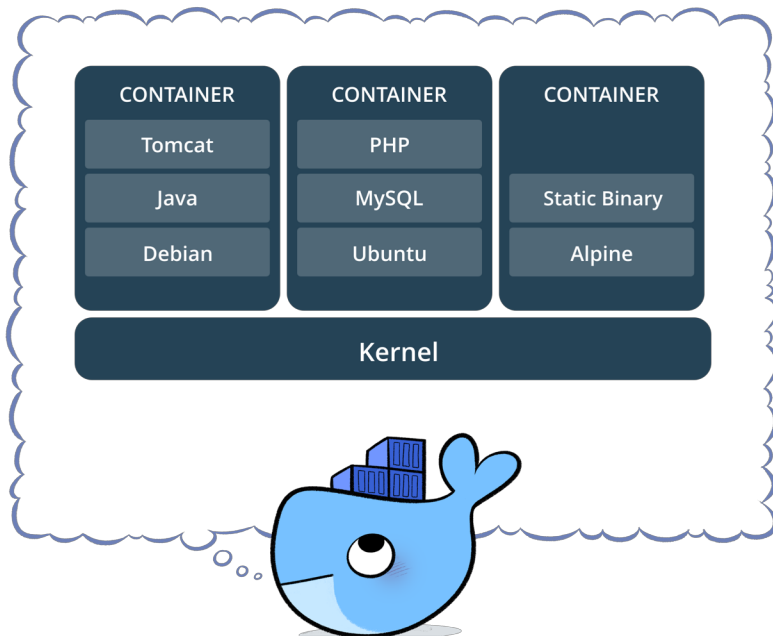
<https://docs.docker.com/engine/installation/>

# INTRODUCCIÓN A DOCKER

## ¿Qué es Docker?

- Docker es un software para la gestión de contenedores

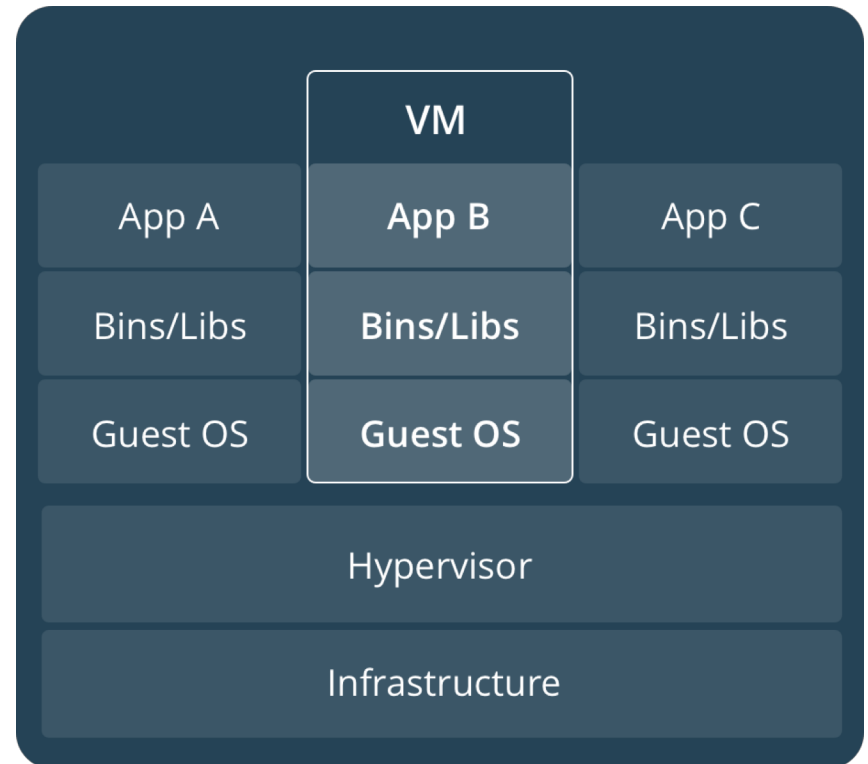
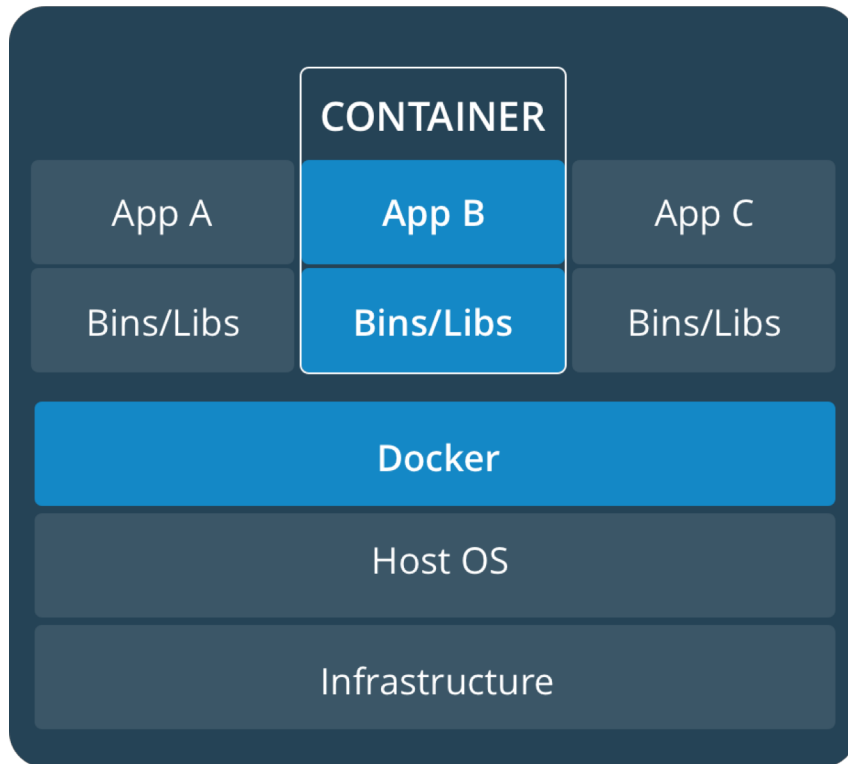




## ¿Qué es un contenedor?

- Una forma de empaquetar software en un formato que incluye todo lo necesario para hacerlo funcionar y se ejecuta aislado del resto de la máquina
- Tiene dos conceptos muy relacionados:
  - La imagen, que es un paquete ejecutable que incluye todo lo necesario para ejecutar un software
  - El contenedor, que es la instancia en ejecución de una imagen, es decir, lo que la imagen

## ¿Y esto no es lo mismo que una máquina virtual?



**CONTAINERS  
ARE  
LINUX.®**



redhat.

# PRIMEROS PASOS



Instala Docker

(si no funciona <http://play-with-docker.com> )

## Nuestro “hello world” con Docker

```
> docker run hello-world
```

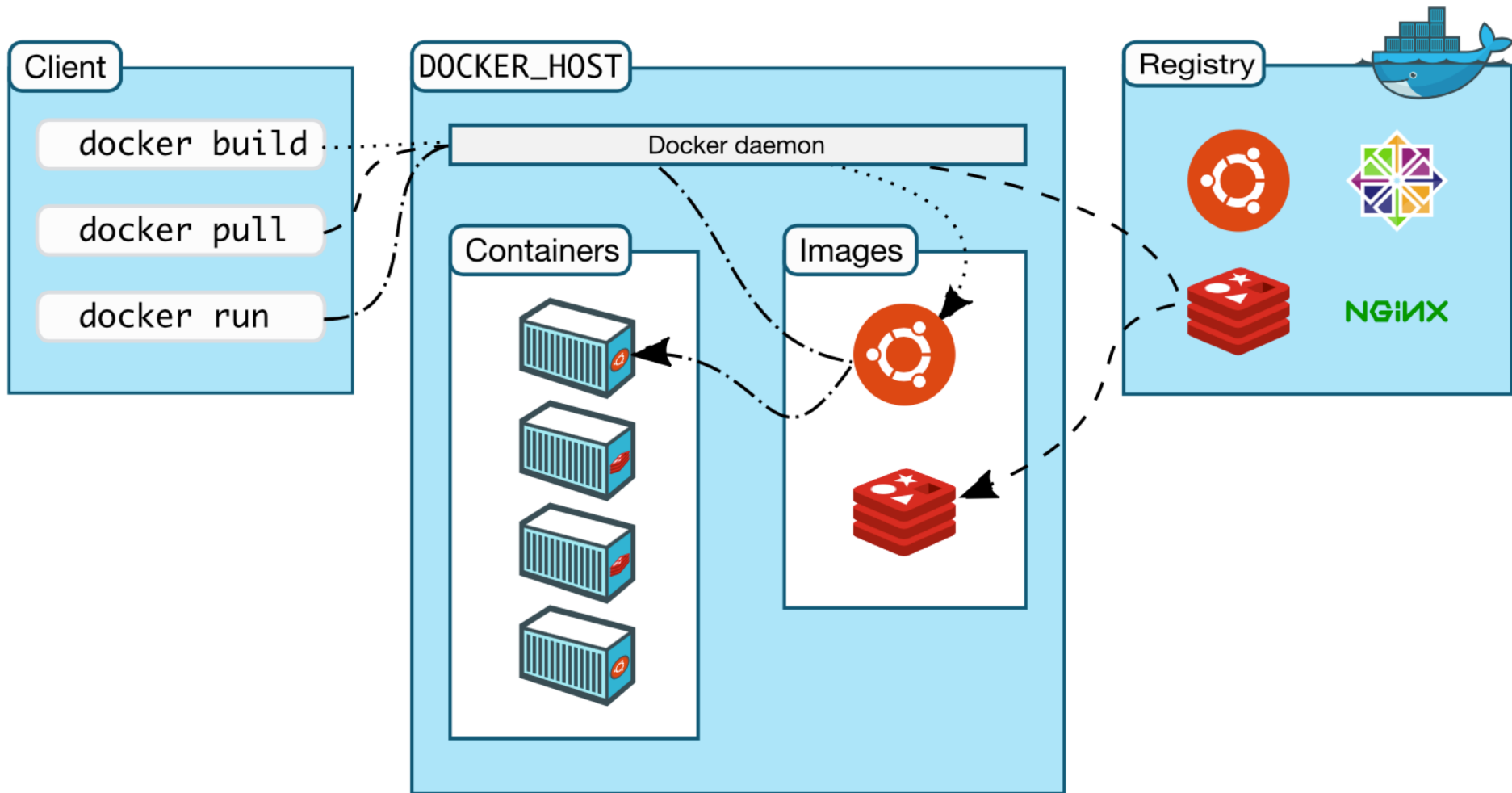
Docker CLI



Lanzar una imagen

Nombre de la imagen

# ¿Qué pasó?




## Otro ejemplo

```
> docker run -i -t ubuntu /bin/bash
```

Interactivo



Comando a ejecutar



## Otro más


Puerto local : Puerto contenedor

```
> docker run -p 8000:80 -d  
kitematic/hello-world-nginx
```

En segundo plano

## Otro más

```
> docker run -p 8010:80 -d -v  
/c/Users/me/Desktop/nginx_files:/website_files  
kitematic/hello-world-nginx
```



En Linux y Mac hay que poner la ruta completa.  
Si se usa Windows 10 hay que poner C:/Users...

Edita el index.html que ha aparecido en nginx\_files y prueba cómo se actualiza dinámicamente

¿Puedo tener más de una máquina?

> docker ps

## Esta es una lista de comandos básicos:

```
docker run -d -p 4000:80 friendlyname#Run "friendlyname" mapping port 4000 to 80
docker container ls # List all running containers
docker container ls -a # List all containers, even those not running
docker container stop <hash> # Gracefully stop the specified container
docker container kill <hash> # Force shutdown of the specified container
docker container rm <hash> # Remove specified container from this machine
docker container rm $(docker container ls -a -q) # Remove all containers
docker image ls -a # List all images on this machine
docker image rm <image id> # Remove specified image from this machine
docker image rm $(docker image ls -a -q) # Remove all images from this machine
docker logs <containerName> # Shows the log of a container
```



# ¿Y de dónde salen las imágenes?

mysql

FILTER E

## Recommended



**official**  
**mysql**

MySQL is a widely used, open-source relational database management system (RDBMS).

♡ 4.1K ↓ 69M    **CREATE**

<https://hub.docker.com>

## Other Repositories



**mysql**  
**mysql-server**

Optimized MySQL Server Docker images. Created, maintained and supported by the MySQL team a...


♡ 284 ↓ 4M    **CREATE**



**tozd**  
**mysql**

MySQL (MariaDB fork) Docker image.


♡ 1 ↓ 2K    **CREATE**



**alterway**  
**mysql**

Docker MySQL


♡ 3 ↓ 1K    **CREATE**



**centurylink**  
**mysql**

Image containing mysql. Optimized to be linked to another image/container.


♡ 49 ↓ 6M    **CREATE**



**cloudposse**  
**mysql**

Improved `mysql` service with support for `mysqld\_safe` and `fixtures` loaded from...

♡ 0 ↓ 540K    **CREATE**



**bitnami**  
**mysql**

Bitnami MySQL Docker Image

♡ 4 ↓ 1K    **CREATE**

# DOCKERIZANDO APLICACIONES

## ¿Qué queremos conseguir?

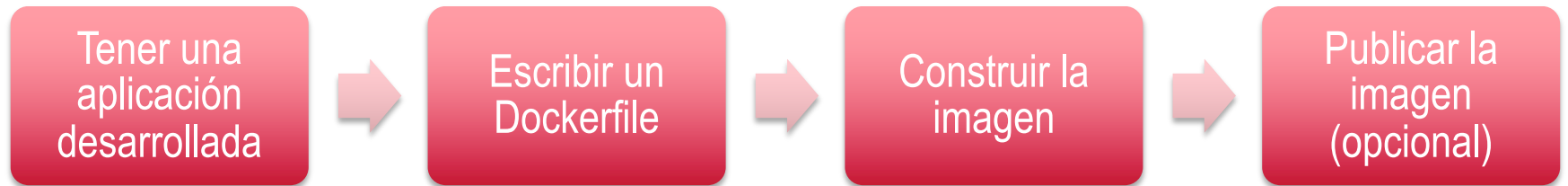
- Tener empaquetada nuestra aplicación y sus dependencias en una imagen para poder desplegarla donde queramos simplemente con

```
> docker run miAplicacion
```

## Imágenes de docker

- Una imagen es una colección de archivos
- Se parte de una imagen base y luego se construyen imágenes personalizadas encima
- Un Dockerfile es un fichero que describe las instrucciones para construir una nueva imagen
- Las imágenes están en capas y cada capa representa un diff de la capa anterior

## Pasos para Dockerizar una aplicación



## Nuestra aplicación

- Un “Hello world” hecho en Java con Spring MVC
  - Entrando en la ruta /greeting dentro de donde esté el WAR desplegado aparece un “Hola, mundo!”
  - Por ejemplo, si el WAR se despliega en hello-java-0.1.0, la ruta es hello-java-0.1.0/greeting/
- Se empaqueta en un WAR

## El Dockerfile

```
# Base image
FROM tomcat:8-jre8

# Copy war to tomcat path
ADD target/hello-java-0.1.0.war /usr/local/tomcat/webapps
```

Consejos para escribir Dockerfiles: [https://docs.docker.com/engine/userguide/eng-image/dockerfile\\_best-practices/](https://docs.docker.com/engine/userguide/eng-image/dockerfile_best-practices/)

## Construimos la imagen y la comprobamos

```
> docker build -t javahello .
```

```
> docker images
```

```
> docker run -it --rm -p 8080:8080 javahello
```



## Publicar nuestra imagen (opcional)

- Hay que registrarse en DockerHub (<http://hub.docker.com> )

> docker login

> docker push mi\_usuario/javahello

# CONCLUSIONES

## ¿Para qué me sirve Docker como desarrollador?

- Entornos de desarrollo:
  - Compartibles
  - Seguros
  - Limpios
  - Extensibles
- Asegura el mismo entorno en:
  - Todos los desarrolladores
  - Pruebas
  - Producción
- Facilita gestionar varias versiones de una misma aplicación

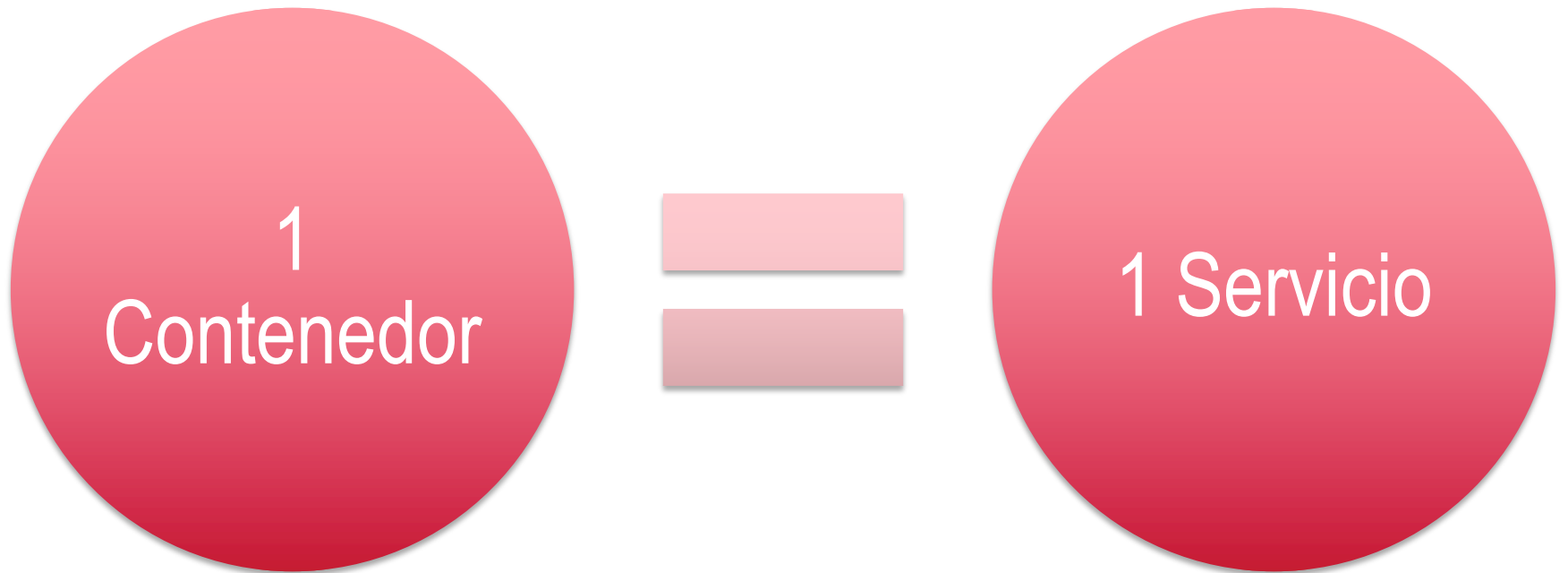
## ¿Para qué me sirve como administrador?

- Despliegue independiente de la tecnología (Java, PHP, NodeJS...)
- Elimina inconsistencias entre entornos de desarrollo, prueba y producción
- Permite desplegar de forma similar en:
  - El portátil del desarrollador
  - En máquinas virtuales en un data center
  - En servidores cloud (AWS, Azure, DigitalOcean...)
  - En una mezcla de ellos
- Ofrece facilidades de escalado y gestión de clusteres

**Y sobre todo...**

Es MUCHO más ligero que una Máquina Virtual

**En Docker se recomienda seguir el principio de responsabilidad única:**



## Docker en la Universidad

- **Regístrate** en <http://dockr.ly/students> y obtendrás:
  - Acceso al *Docker Student Kit*.
  - Últimas **novedades** y actualizaciones sobre Docker.
  - **Invitaciones** y códigos de **descuento** a **eventos** de Docker para estudiantes.
  - Posibilidad de conseguir **acceso prioritario** a betas y lanzamientos de productos.
  - Oportunidad de convertirte en *Docker Ambassador*.
  - Acceso al **canal de Slack** de Docker (#docker-students).

# Recursos

- **Cursos:**

- Laboratorios virtuales gratuitos: <http://training.play-with-docker.com/>
- Cursos gratuitos oficiales: <http://training.docker.com/category/self-paced-online>

- **Libros:**

- Docker Cookbook: <http://shop.oreilly.com/product/0636920036791.do>
- Using Docker: <http://shop.oreilly.com/product/0636920035671.do>
- Docker: Up & Running: <http://shop.oreilly.com/product/0636920036142.do>



## Agradecimientos

- Parte de estas transparencias están muy inspiradas (incluso copiadas) de una presentación de Docker de Antonio Gámez (<http://personal.us.es/agamez2/conferencias/docker-y-kubernetes-el-futuro-de-la-distribucion-de-aplicaciones-en-la-nube/> )