

#PyTatuy2016

Usando Docker con Django

Ernesto Crespo

Blog: <http://blog.crespo.org.ve>

Twitter: https://twitter.com/_seraph1





Usando Docker con Django por [Ernesto Crespo](#) se distribuye bajo una [Licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional](#).

Basada en una obra en <http://www.slideshare.net/ecrespo/usando-django-con-docker>.

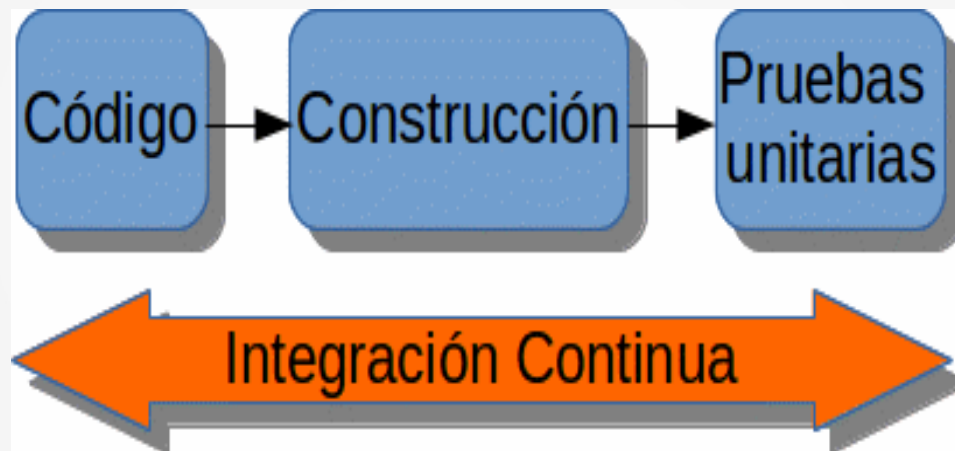
#PyTatuy2016



Agenda

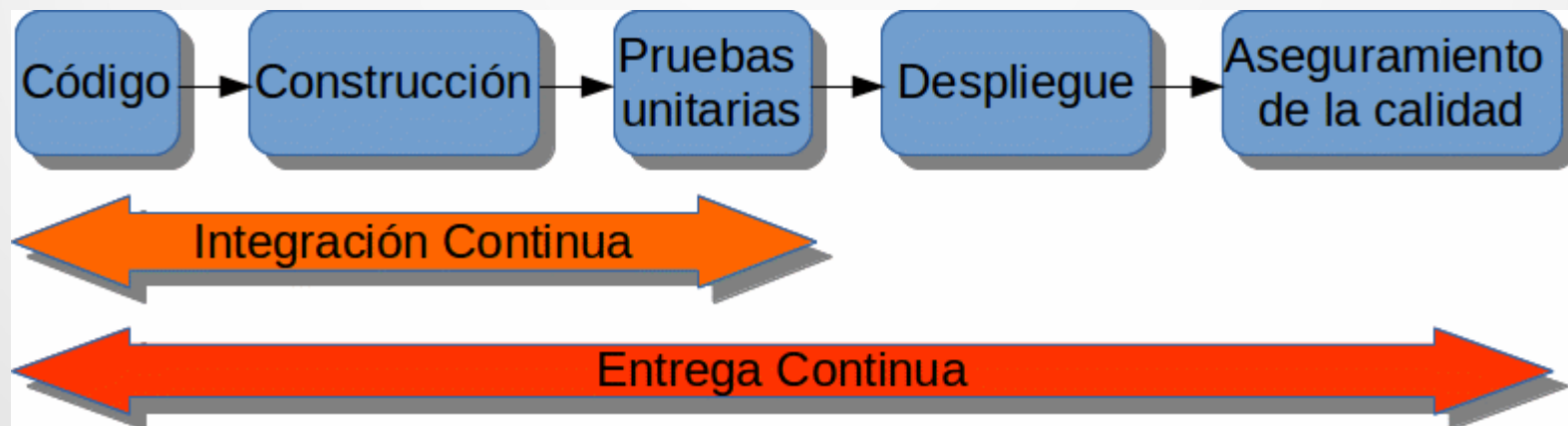
- Integración continua, entrega continua y despliegue continuo
- Esquema
- DevOps
- Docker
 - Contenedores vs Máquinas virtuales
 - Contenedores
 - Docker como ambiente de desarrollo
 - Docker en entrega continua
 - Imágenes y contenedores
- Django
 - Admin de Django en un contenedor
 - Blog del Tutorial de DjangoGirl en un contenedor
 - Microservicios (postgresql+Django)

Integración Continua



Entrega continua

Es un de Ingeniería de Software en el que los equipos mantienen la producción de software en ciclos cortos y se asegura de que el software puede liberarse de forma fiable en cualquier momento

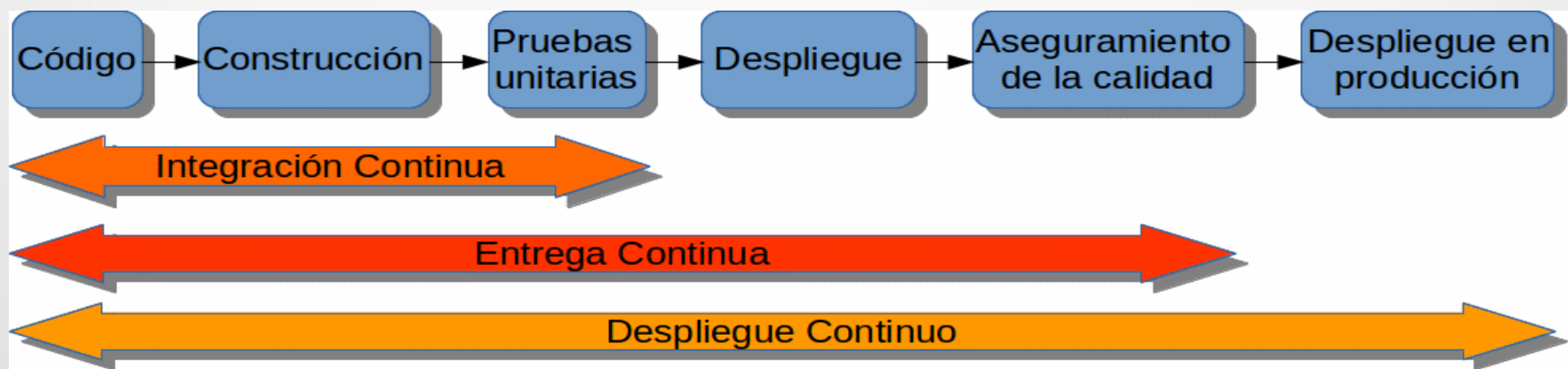


Despliegue Continuo

Es la implementación del patrón Fail Fast (Fallar rápido) para un proceso de entrega de software.

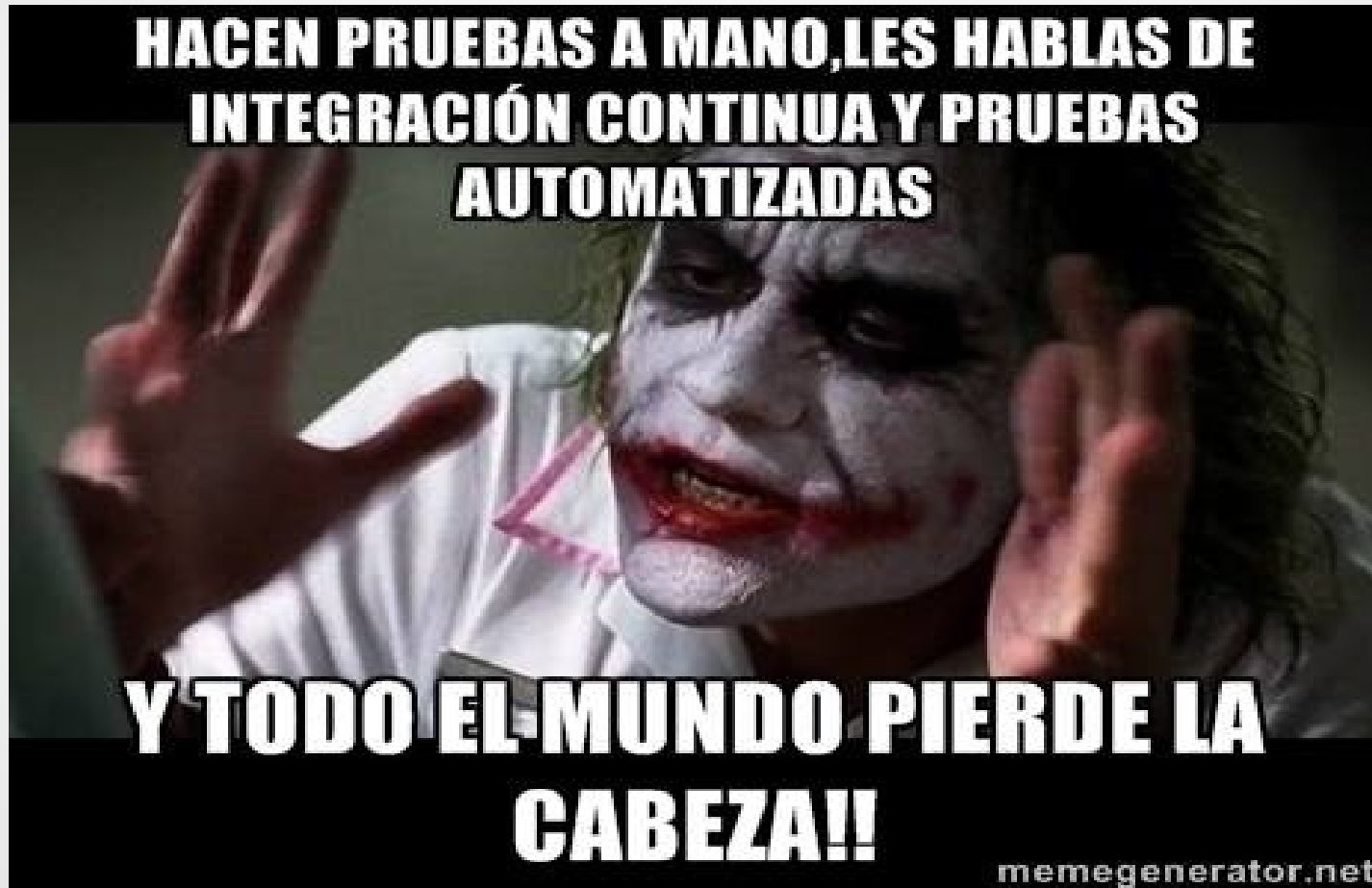
Mientras más cerca ocurra un error del punto en el que fue introducido, más datos vamos a tener para corregir dicho error.

Cada cambio que supera de manera exitosa las pruebas automáticas podemos desplegarlo de forma automática en producción

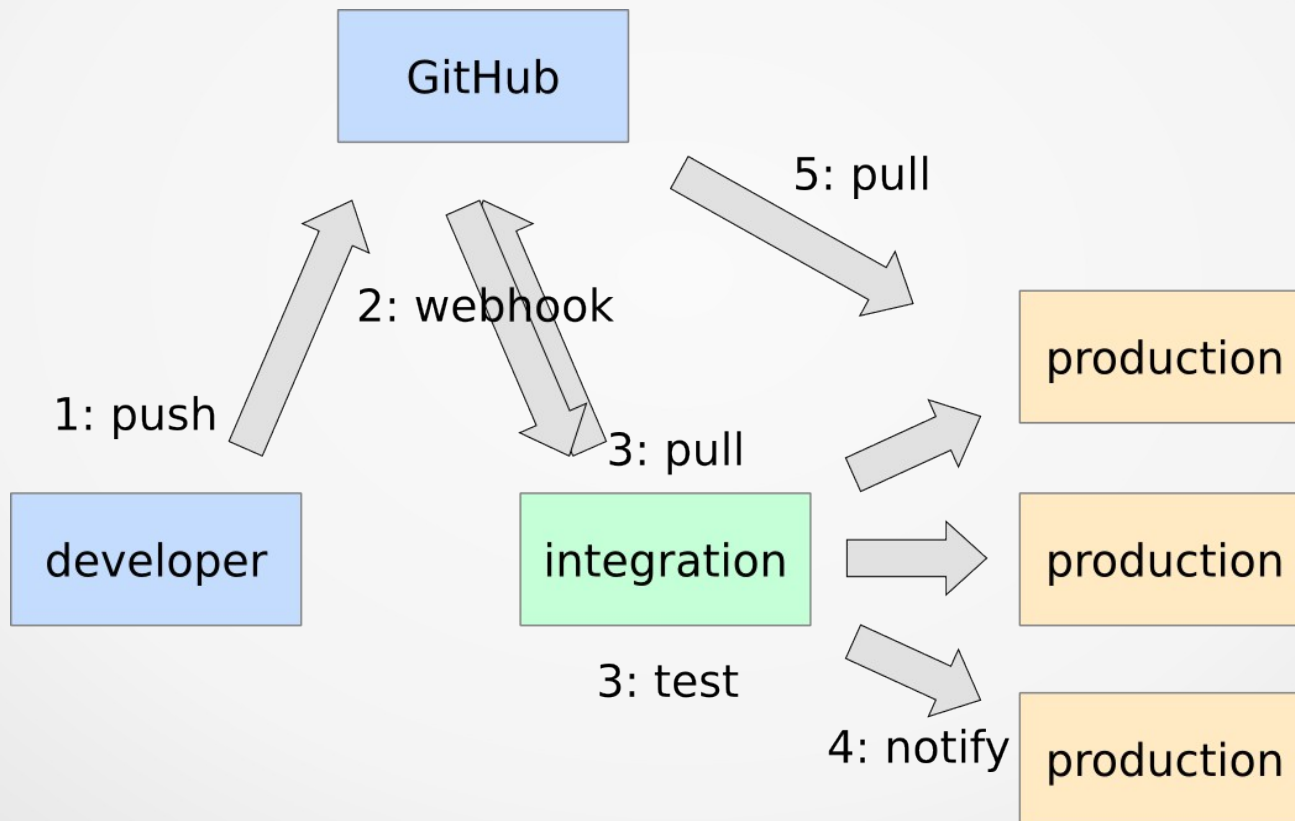


#PyTatuy2016

Despliegue Continuo



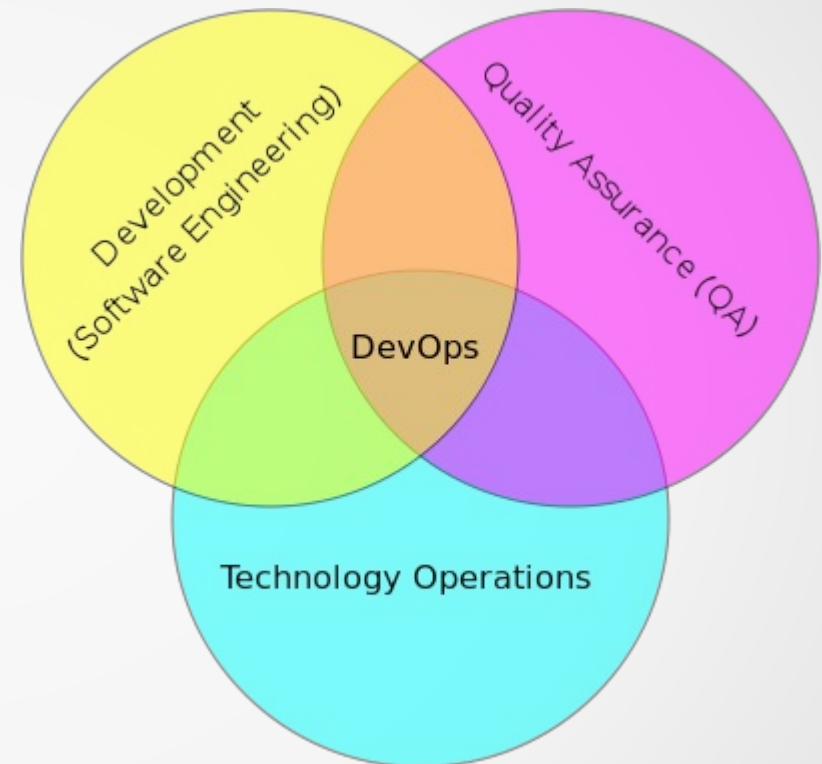
Esquema



DevOps

Acrónimo en Inglés de Development (desarrollo y operations(operaciones), se refiere a la cultura o movimiento que se centra en la comunicación, colaboración e integración entre desarrolladores de software y los profesionales de operaciones en TIC.

DevOps es una respuesta a la interdependencia del desarrollo de software y las operaciones TIC.



#PyTatuy2016

Docker

Implementación de alto nivel para proporcionar contenedores livianos que ejecutan procesos aislados



docker

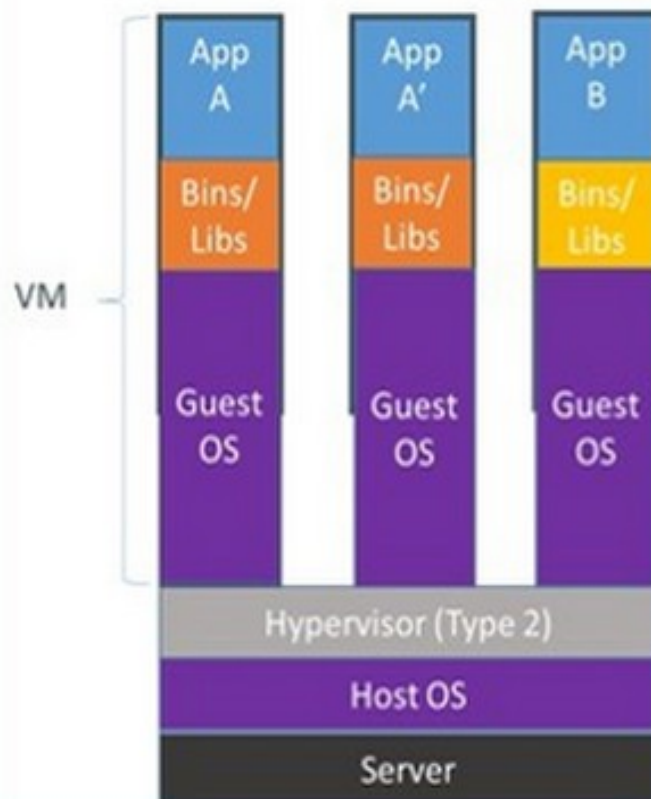
#PyTatuy2016

Docker

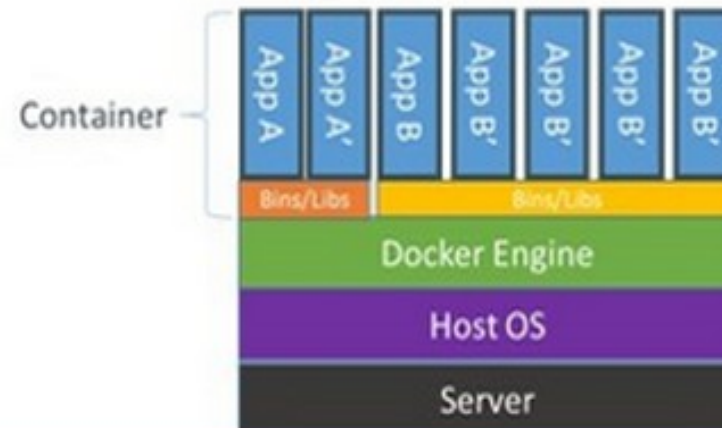


Docker

Containers vs. VMs

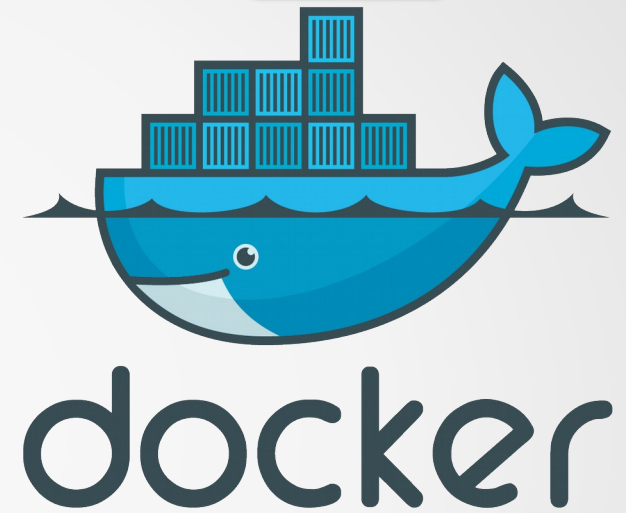
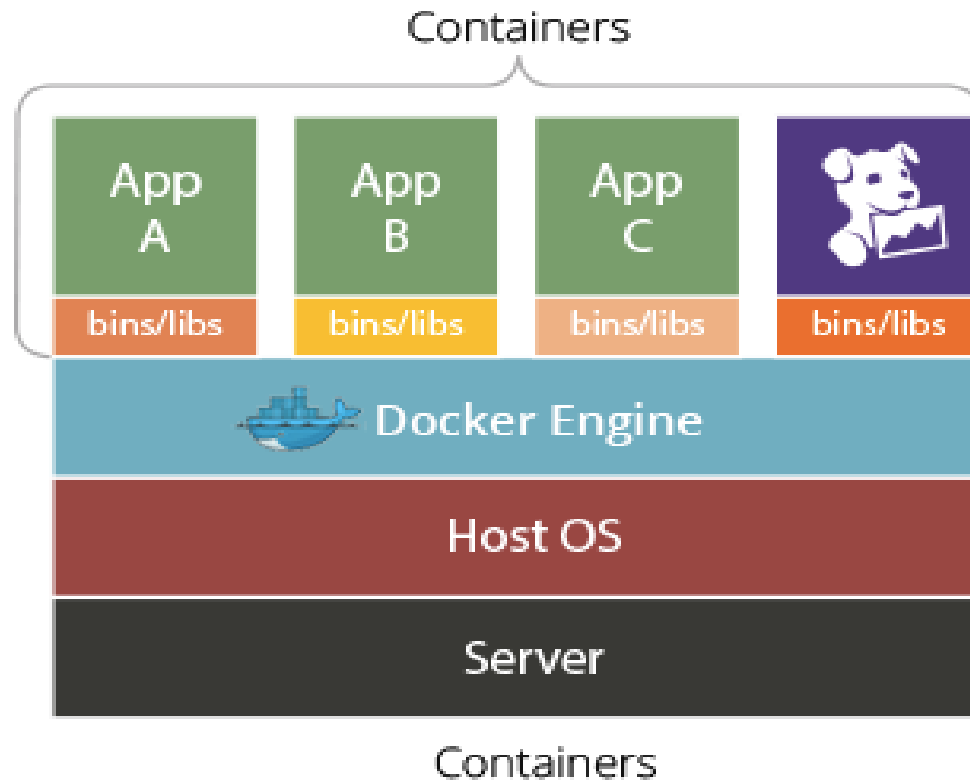


Containers are isolated, but share OS and, where appropriate, bins/libraries



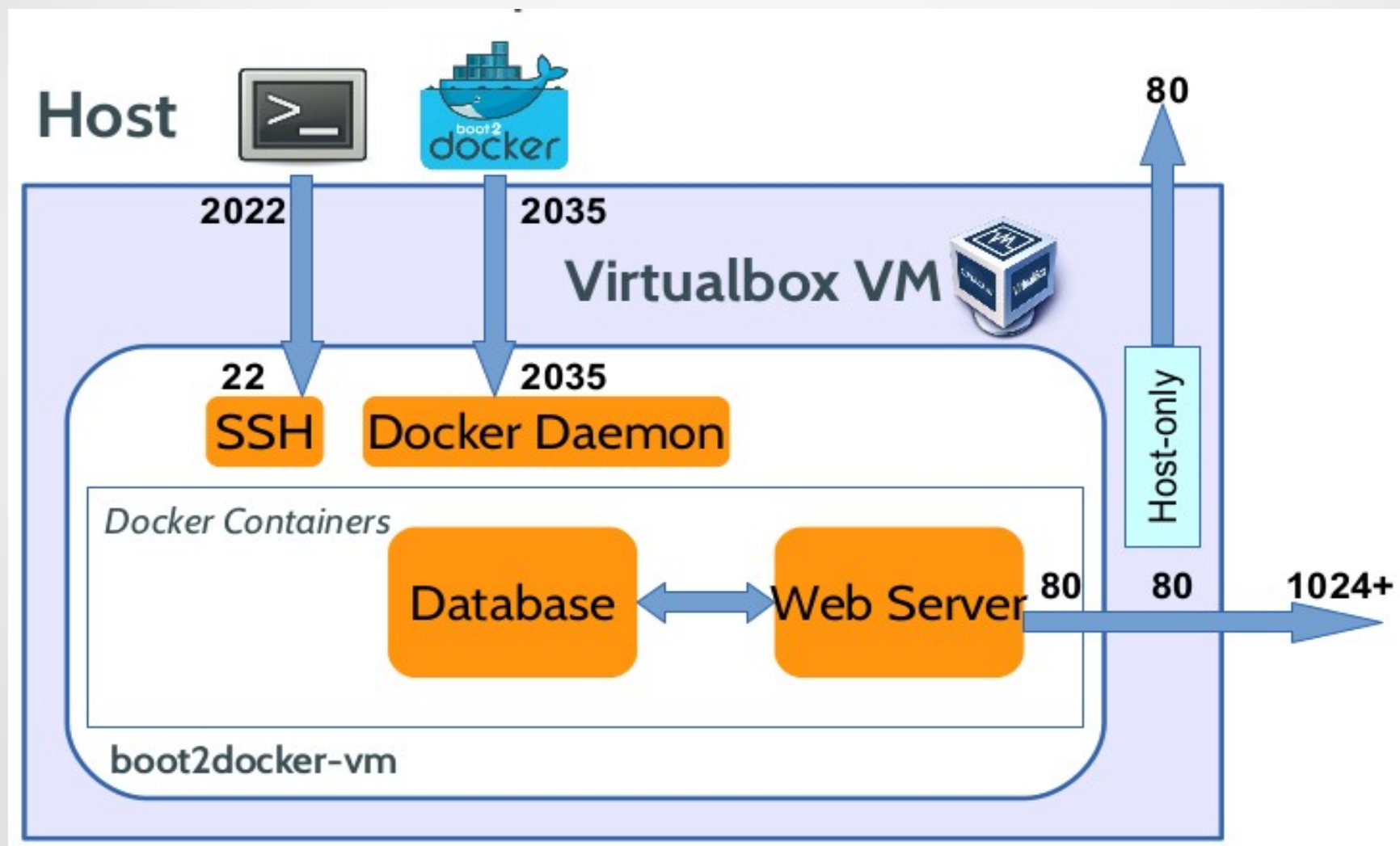
#PyTatuy2016

Docker (contenedores)



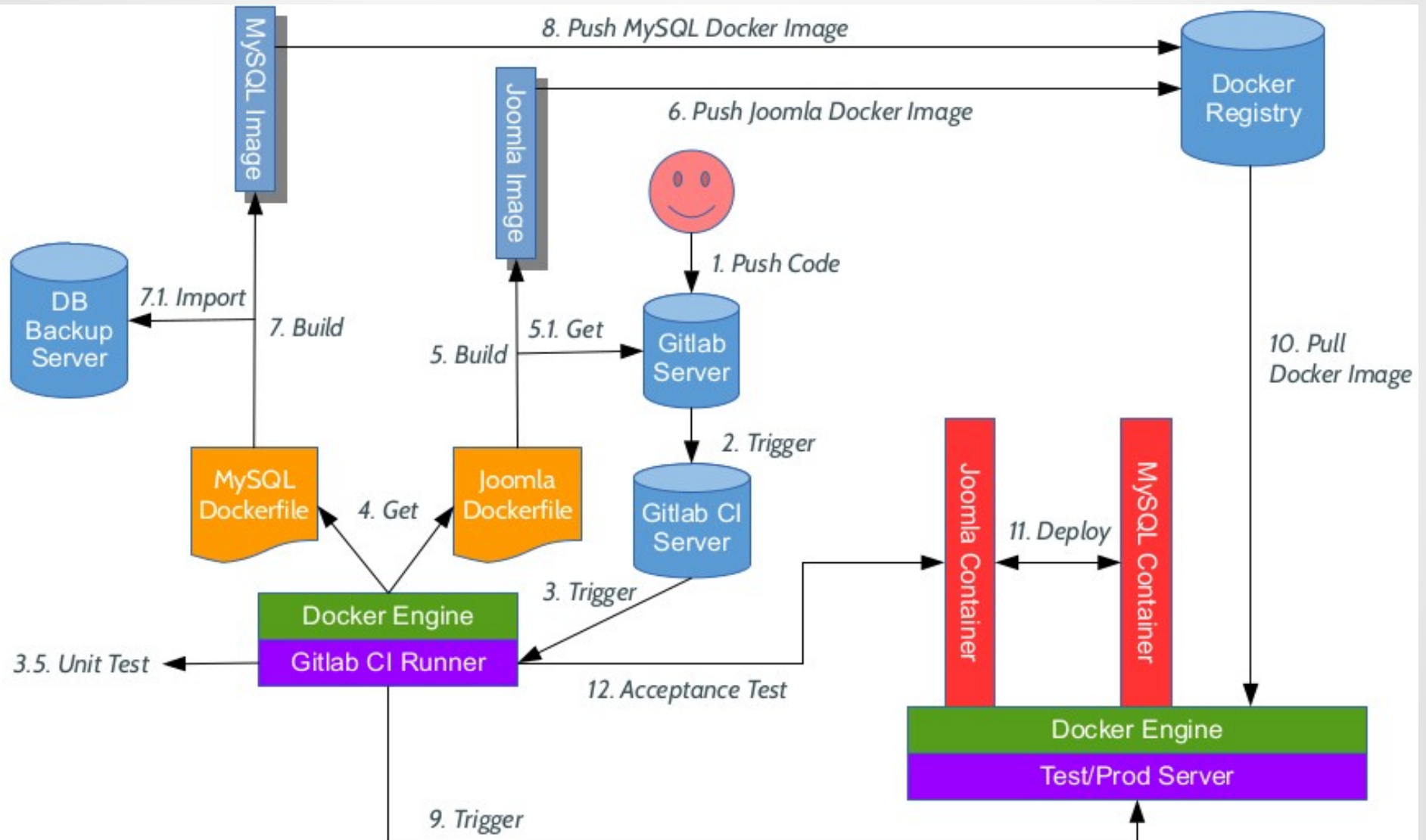
#PyTatuy2016

Docker como ambiente de desarrollo



#PyTatuy2016

Docker en entrega continua (ejm de joomla)



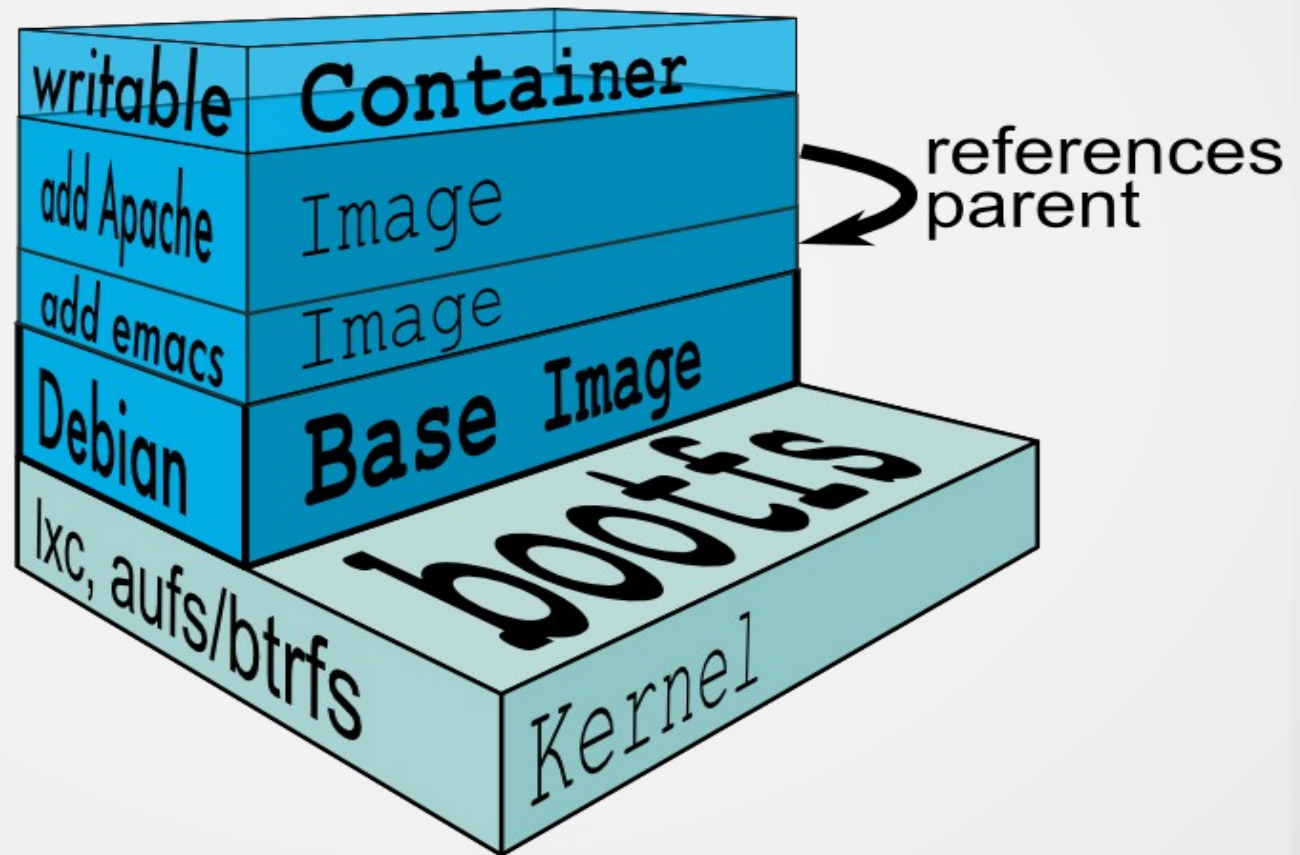
#PyTatuy2016

Docker en entrega continua



#PyTatuy2016

Docker Imágenes y contenedores



#PyTatuy2016

Instalación y comandos básicos:

Instalación de docker:

<http://blog.crespo.org.ve/2015/12/instalar-docker-en-debian-jessie.html>

Uso de Docker en Debian Jessie Parte 1:

<http://blog.crespo.org.ve/2015/12/uso-de-docker-en-debian-jessie-parte-1.html>

Uso de Docker en Debian Jessie Parte 2:

<http://blog.crespo.org.ve/2015/12/uso-de-docker-en-debian-jessie-parte-2.html>

Crear una imagen Docker a partir de un archivo Dockerfile:

<http://blog.crespo.org.ve/2016/01/crear-una-imagen-docker-partir-de-un.html>

Ejecutando microservicios con Docker usando docker-compose:

<http://blog.crespo.org.ve/2016/01/ejecutando-microservicios-con-docker.html>

#PyTatuy2016

Instalación y comandos básicos:

Con docker se puede tener:

- Ambiente de desarrollo limpio y de rápida recuperación
- Ambiente de pruebas
- Ambiente de integración y entrega

Iniciando el Admin de Django con Docker

Se tiene un directorio de proyecto Django con el nombre djangoapp:

```
djangoapp
├── db.sqlite3
├── djangoapp
│   ├── __init__.py
│   ├── __init__.pyc
│   ├── settings.py
│   ├── settings.pyc
│   ├── urls.py
│   ├── urls.pyc
│   ├── wsgi.py
│   └── wsgi.pyc
└── manage.py
```

Iniciando el Admin de Django con Docker

Se tiene un directorio django con el contenido de djangoapp y el archivo Dockerfile:

```
django
├── djangoapp
│   ├── db.sqlite3
│   ├── djangoapp
│   │   ├── __init__.py
│   │   ├── __init__.pyc
│   │   ├── settings.py
│   │   ├── settings.pyc
│   │   ├── urls.py
│   │   ├── urls.pyc
│   │   ├── wsgi.py
│   │   └── wsgi.pyc
│   └── manage.py
└── Dockerfile
```

Iniciando el Admin de Django con Docker

Se tiene un archivo llamado Dockerfile con lo siguiente:

```
1 FROM debian
2 MAINTAINER Ernesto Crespo
3 RUN apt-get update
4 RUN apt-get upgrade -y --force-yes
5 RUN apt-get install -y python python-pip
6 RUN apt-get clean
7 RUN pip install django
8
9 # Copiar aplicacion del subdirectorio django_app/ al directorio
10 # /djangoapp en el contenedor
11 ADD djangoapp /srv/djangoapp
12 COPY djangoapp /srv/djangoapp
13 RUN chown -R www-data:www-data /srv/djangoapp
14 RUN chmod a+x /srv/djangoapp/manage.py
15 # Establecer el directorio de trabajo
16 WORKDIR /srv/djangoapp
17
18
19
20
21 EXPOSE 8000
22
23 ENTRYPOINT python /srv/djangoapp/manage.py runserver &
24
25
```


#PyTatuy2016

Iniciando el Admin de Django con Docker

Se construye la imagen con el comando:
`docker build -t myapp .`

Listar las imagenes:

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
myapp	latest	1fd728b9a9da	3 days ago	401.6 MB
debian	latest	8b9a99209d5c	9 weeks ago	125.1 MB
sonarqube	latest	dd47274097f7	3 months ago	942.5 MB
debian	8.1	f05335696a9b	5 months ago	125.2 MB

#PyTatuy2016

Iniciando el Admin de Django con Docker

Iniciando un contenedor de Docker:

```
docker run -p 8000:8000 myapp
```

Se lista los procesos de Docker

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
61173c48c8a9	myapp	"/bin/sh -c 'python m"	25 seconds ago	Up 18 seconds	0.0.0.0:8000->8000/tcp	sad_mcclintock

#PyTatuy2016

Iniciando el Admin de Django con Docker



#PyTatuy2016

Iniciando el Admin de Django con Docker

Se puede arrancar una consola interactiva del contenedor django:

```
docker exec -it sad_mcclintock /bin/bash
```

```
root@61173c48c8a9:/srv/djangoapp#
```

Desde allí se puede crear un usuario:

```
python manage.py createsuperuser --username=admin --email=admin@myapp.com
```

Password:

Password (again):

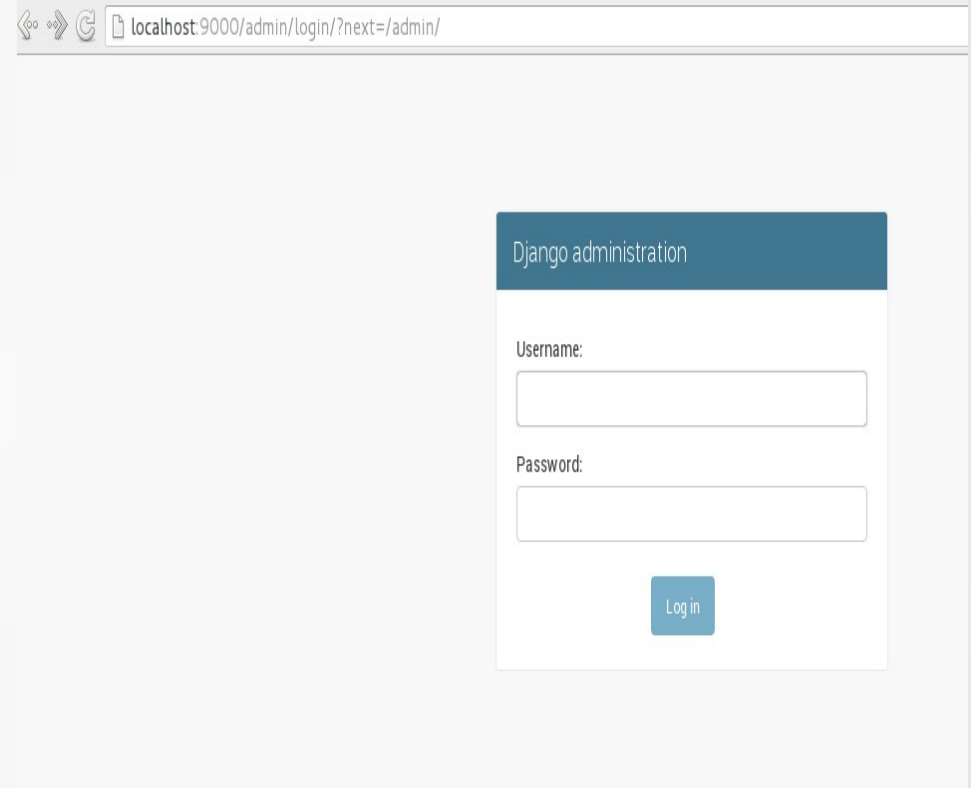
Superuser created successfully.

#PyTatuy2016

Iniciando el Admin de Django con Docker

Se puede iniciar otro contenedor para visualizarlo en un puerto diferente:

```
docker run -p 9000:8000 myapp
Not Found: /admin
[11/Feb/2016 13:53:01] "GET /admin HTTP/1.1" 301 0
[11/Feb/2016 13:53:01] "GET /admin/ HTTP/1.1" 302 0
[11/Feb/2016 13:53:01] "GET /admin/login/?
next=/admin/ HTTP/1.1" 200 1684
[11/Feb/2016 13:53:02] "GET
/static/admin/css/base.css HTTP/1.1" 200 15897
[11/Feb/2016 13:53:02] "GET
/static/admin/css/login.css HTTP/1.1" 200 1203
[11/Feb/2016 13:53:02] "GET
/static/admin/css/fonts.css HTTP/1.1" 200 423
[11/Feb/2016 13:53:02] "GET
/static/admin/fonts/Roboto-Light-webfont.woff
HTTP/1.1" 200 81348
[11/Feb/2016 13:53:02] "GET
/static/admin/fonts/Roboto-Regular-webfont.woff
HTTP/1.1" 200 80304
Not Found: /favicon.ico
[11/Feb/2016 13:53:02] "GET /favicon.ico HTTP/1.1"
404 2674
```



localhost:9000/admin/login/?next=/admin/

Django administration

Username:

Password:

Log in

#PyTatuy2016

Un ejemplo más completo – Tutorial de Django Girl

```
django3
├── djangoapp
│   ├── blog
│   │   ├── admin.py
│   │   ├── admin.pyc
│   │   ├── apps.py
│   │   ├── forms.py
│   │   ├── forms.pyc
│   │   ├── __init__.py
│   │   ├── __init__.pyc
│   │   ├── migrations
│   │   │   ├── 0001_initial.py
│   │   │   ├── 0001_initial.pyc
│   │   │   ├── __init__.py
│   │   │   └── __init__.pyc
│   │   ├── models.py
│   │   ├── models.pyc
│   │   ├── static
│   │   │   └── css
│   │   │       └── blog.css
│   │   ├── templates
│   │   │   └── blog
│   │   │       ├── base.html
│   │   │       ├── post_detail.html
│   │   │       ├── post_edit.html
│   │   │       └── post_list.html
│   │   ├── tests.py
│   │   ├── urls.py
│   │   ├── urls.pyc
│   │   ├── views.py
│   │   └── views.pyc
│   ├── db.sqlite3
│   ├── djangoapp
│   │   ├── __init__.py
│   │   ├── __init__.pyc
│   │   ├── settings.py
│   │   ├── settings.pyc
│   │   ├── urls.py
│   │   ├── urls.pyc
│   │   ├── wsgi.py
│   │   └── wsgi.pyc
│   └── manage.py
```

Un ejemplo más completo – Tutorial de Django Girl

```
├── static
│   ├── admin
│   │   ├── css
│   │   │   ├── base.css
│   │   │   ├── changelists.css
│   │   │   ├── dashboard.css
│   │   │   ├── fonts.css
│   │   │   ├── forms.css
│   │   │   ├── login.css
│   │   │   ├── rtl.css
│   │   │   └── widgets.css
│   │   ├── fonts
│   │   │   ├── LICENSE.txt
│   │   │   ├── README.txt
│   │   │   ├── Roboto-Bold-webfont.woff
│   │   │   ├── Roboto-Light-webfont.woff
│   │   │   └── Roboto-Regular-webfont.woff
│   │   └── img
│   │       ├── calendar-icons.svg
│   │       ├── gis
│   │       │   ├── move_vertex_off.svg
│   │       │   └── move_vertex_on.svg
│   │       ├── icon-addlink.svg
│   │       ├── icon-alert.svg
│   │       ├── icon-calendar.svg
│   │       ├── icon-changelink.svg
│   │       ├── icon-clock.svg
│   │       ├── icon-deletelink.svg
│   │       ├── icon-no.svg
│   │       ├── icon-unknown-alt.svg
│   │       ├── icon-unknown.svg
│   │       ├── icon-yes.svg
│   │       ├── inline-delete.svg
│   │       ├── LICENSE
│   │       ├── README.txt
│   │       ├── search.svg
│   │       ├── selector-icons.svg
│   │       ├── sorting-icons.svg
│   │       ├── tagadd.svg
│   │       └── tagarrowright.svg
```

#PyTatuy2016

Un ejemplo más completo – Tutorial de Django Girl

```
├── js
│   ├── actions.js
│   ├── actions.min.js
│   ├── admin
│   │   ├── DateTimeShortcuts.js
│   │   └── RelatedObjectLookups.js
│   ├── calendar.js
│   ├── collapse.js
│   ├── collapse.min.js
│   ├── core.js
│   ├── inlines.js
│   ├── inlines.min.js
│   ├── jquery.init.js
│   ├── prepopulate.js
│   ├── prepopulate.min.js
│   ├── SelectBox.js
│   ├── SelectFilter2.js
│   ├── timeparse.js
│   ├── urlify.js
│   ├── vendor
│   │   ├── jquery
│   │   │   ├── jquery.js
│   │   │   ├── jquery.min.js
│   │   │   └── LICENSE-JQUERY.txt
│   │   └── xregexp
│   │       ├── LICENSE-XREGEXP.txt
│   │       └── xregexp.min.js
└── Dockerfile
```

#PyTatuy2016

Un ejemplo más completo – Tutorial de Django Girl

El archivo Dockerfile contiene:

```
1 FROM debian
2 MAINTAINER Ernesto Crespo <ecrespo@gmail.com>
3 RUN apt-get update
4 RUN apt-get install -y python-pip
5 RUN apt-get clean
6 RUN pip install virtualenv
7
8 RUN pip install django
9
10 RUN echo "America/Venezuela/Caracas" > /etc/timezone && dpkg-reconfigure -f noninteractive
    tzdata
11
12 # Exposing the APP at port 8000
13 EXPOSE 8000
14
15 # Copiar aplicacion del subdirectorio django_app/ al directorio
16 # /djangoapp en el contenedor
17 ADD djangoapp /srv/djangoapp
18 COPY djangoapp /srv/djangoapp
19 RUN chown -R www-data:www-data /srv/djangoapp
20 RUN chmod a+x /srv/djangoapp/manage.py
21 # Establecer el directorio de trabajo
22 WORKDIR /srv/djangoapp
23
24
25 # Install provisioning dependencies
26 #RUN django-admin.py startproject huesped .
27 #RUN python manage.py migrate
28
29 # Launching MyApp project
30 CMD python manage.py runserver 0.0.0.0:8000
```


#PyTatuy2016

Un ejemplo más completo – Tutorial de Django Girl

Se corre el contenedor:

```
docker run -p 8000:8000 django3
```

localhost:8000

Blog Post

prueba 2

esta es otra prueba

4 de February de 2016 a las 14:27

Prueba1

Esta es una prueba.

Guardar

4 de February de 2016 a las 15:03

Otro post

probando crear un post

agregado

4 de February de 2016 a las 15:13

de nuevo probando

4 de February de 2016 a las 15:13

Docker con microservicios usando docker-compose

Archivo Dockerfile:

```
1 FROM python:2.7
2 ENV PYTHONUNBUFFERED 1
3 RUN mkdir /app
4 WORKDIR /app
5 ADD requerimientos.txt /app/
6 RUN pip install --upgrade pip
7 RUN pip install -r requerimientos.txt
8 ADD . /app/
9
10
```

Archivo requerimientos.txt:

```
1 Django
2 psycopg2
3
4
```

#PyTatuy2016

Docker con microservicios usando docker-compose

Archivo docker-compose.yml:

```
1 db:
2   image: postgres
3 web:
4   build: .
5   command: python manage.py runserver 0.0.0.0:8000
6   volumes:
7     - .:/app
8   ports:
9     - "8000:8000"
10  links:
11    - db
12
13
```

#PyTatuy2016

Docker con microservicios usando docker-compose

Se ejecuta docker-compose iniciando un proyecto Django:
docker-compose run web django-admin.py startproject djangoapp .

Al terminar se tiene lo siguiente:

```
drwxr-xr-x 2 root  root  4096 feb 10 21:35 djangoapp
-rw-r--r-- 1 ernesto ernesto 174 feb 10 21:14 docker-compose.yml
-rw-r--r-- 1 ernesto ernesto 177 feb 10 21:14 Dockerfile
-rwxr-xr-x 1 root  root  252 feb 10 21:35 manage.py
-rw-r--r-- 1 ernesto ernesto 17 feb 10 20:10 requerimientos.txt
```

Y la imagen:

docker images
REPOSITORY
VIRTUAL SIZE

TAG

IMAGE ID

CREATED

docker5_web

latest

7a7faf87cf21

4 minutes ago

715.2 MB

postgres

latest

54fa18d9f3b6

2 weeks ago

263.8 MB

Docker con microservicios usando docker-compose

Se modifica el archivo `djangoapp/settings.py`, el contenedor de postgresql tiene usuario postgres base de datos postgres y host db:

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.postgresql_psycopg2',  
        'NAME': 'postgres',  
        'USER': 'postgres',  
        'HOST': 'db',  
        'PORT': 5432,  
    }  
}
```

Docker con microservicios usando docker-compose

Se ejecuta migrate de manage.py:

```
docker-compose run web python manage.py migrate
```

Operations to perform:

Apply all migrations: admin, contenttypes, auth, sessions

Running migrations:

Rendering model states... DONE

Applying contenttypes.0001_initial... OK

Applying auth.0001_initial... OK

Applying admin.0001_initial... OK

Applying admin.0002_logentry_remove_auto_add... OK

Applying contenttypes.0002_remove_content_type_name... OK

Applying auth.0002_alter_permission_name_max_length... OK

Applying auth.0003_alter_user_email_max_length... OK

Applying auth.0004_alter_user_username_opts... OK

Applying auth.0005_alter_user_last_login_null... OK

Applying auth.0006_require_contenttypes_0002... OK

Applying auth.0007_alter_validators_add_error_messages... OK

Applying sessions.0001_initial... OK

#PyTatuy2016

Docker con microservicios usando docker-compose

Se crea una cuenta de administrador:

```
docker-compose run web python manage.py createsuperuser
```

Username (leave blank to use 'root'): admin

Email address:

Password:

Password (again):

Superuser created successfully.

#PyTatuy2016

Docker con microservicios usando docker-compose

Se ejecuta el contenedor:

```
docker-compose up
```

```
docker5_db_1 is up-to-date
```

```
Creating docker5_web_1
```

```
Attaching to docker5_db_1, docker5_web_1
```

```
db_1 | The files belonging to this database system will be owned by user "postgres".
```

```
db_1 | This user must also own the server process.
```

```
web_1 |
```

```
.....
```

```
web_1 | System check identified no issues (0 silenced).
```

```
web_1 | February 11, 2016 - 12:38:56
```

```
web_1 | Django version 1.9.2, using settings 'djangoapp.settings'
```

```
web_1 | Starting development server at http://0.0.0.0:8000/
```

```
web_1 | Quit the server with CONTROL-C.
```


#PyTatuy2016

Docker con microservicios usando docker-compose

Se abre el navegador en <http://127.0.0.1/admin>



Referencias:

- Pruebas automáticas y despliegue continuo:
<http://slides.com/alexfernandez/pruebas-automaticas-y-despliegue-continuo/fullscreen#/>
- De la nada al “Todo continuo” y del Scrum más canónico al Kanban con #NoEstimate:
<http://davidfergon.github.io/2015-06-16-de-la-nada-al-todo-continuo-de-scrum-a-kanban-y-noestimates.html>
- Agilidad... piensa diferente, pon las cosas al revés, rompe viejos esquemas:
http://www.javiergarzas.com/2016/02/agilidad-piensa-diferente-pon-las-cosas-al-reves-rompe-viejos-esquemas.html?utm_content=buffer464d9&utm_medium=social&utm_source=facebook.com&utm_campaign=buffer
- ¿Tardarais mucho en pasar a producción un cambio en sólo una línea de código? Aprende entrega continua
<http://www.javiergarzas.com/2012/11/entrega-continua-continuous-delivery.html>
- Joomla Continuous Delivery with Docker:
<http://www.slideshare.net/winggundamth/joomla-continuous-delivery-with-docker>
- Artículos sobre docker en mi blog:
<http://blog.crespo.org.ve/search/label/Docker>
- Docker como la máxima expresión de DevOps:
http://www.slideshare.net/gbrey/docker-como-la-maxima-expresion-de-devops-wisit-2015?qid=667529f1-1dda-420c-8df5-7fdfb3195318&v=&b=&from_search=23
- Tutorial de Django Girl:
<http://tutorial.djangogirls.org/es/>
- Introducción a Docker:
<https://juliomunoz.wordpress.com/2015/01/20/introduccion-a-docker/>
- Docker for dummies:
<http://www.adictosaltrabajo.com/tutoriales/docker-for-dummies/>

#PyTatuy2016

¿Preguntas?

"Si he logrado ver más lejos, ha sido porque he subido a hombros de gigantes".
Isaac Newton



Descarga de la presentación: <http://www.slideshare.net/ecrespo/usando-django-con-docker>

Blog: <http://blog.crespo.org.ve/search/label/Docker>