Brounie

MANUAL DE DJANGO STACK: CELERY, SUPERVISOR, GUNICORN, RABBITMQ VIRTUALENV, NGINX Y MYSQL

1. Actualizar los paquetes de Ubuntu o del sistema Debian:

```
# Actualizar el índice local de paquetes
$ sudo apt-get update

# Actualizar todos los paquetes que puedan ser actualizados
$ sudo apt-get dist-upgrade

# Remover los paquetes que no sean necesarios
$ sudo apt-get autoremove

# Reiniciar la maquina (solo necesario para algunas
actualizaciones)
$ sudo reboot
```

2. Instalar build-essential y pythin dev

Build-essential es el paquete que provee todas las herramientas de compilación estándar de C.

Python-dev provee los archivos necesarios para compilar módulos Python/C.

```
$ sudo apt-get install build-essential python-dev
```

3. Instalar Distribute y pip

(herramientas para instalar paquetes de Python)

```
# Descargar distribute
```

```
$ curl -O http://python-distribute.org/distribute_setup.py

# Instalar distribute
$ sudo python distribute_setup.py

# Remover archivos de instalación
$ rm distribute*

# Usar distribute para instalar pip
$ sudo easy_install pip
```

4. Instalar *virtualenv* y *virtualenvwrapper*

Virtualenv se usa para aislar en ambientes virtuales diferentes los paquetes que vamos a instalar.
Virtualenvwrapper se usa para agilizar virtualenv.

```
# Instalar virtualenv y virtualenvwrapper
$ sudo pip install virtualenv virtualenvwrapper

# Editar el archivo .bashrc con ayuda de vim
$ vim .bashrc

# Agregar la siguiente linea al final del archivo para #habilitar
el virtualenvwrapper
source /usr/local/bin/virtualenvwrapper.sh

#Salvar y cerrar el editor

# Salir y volver a acceder
$ exit
```

5. Crear un ambiente virtual

```
# Crear un ambiente virtual
$ mkvirtualenv <NOMBRE_DEL_AMBIENTE_VIRTUAL>
# Algunos comandos útiles para el ambiente virtual:
     # Desactivar ambiente virtual
     $ deactivate
     # Activar ambiente virtual o cambiar a otro
     $ workon <NOMBRE_DEL_AMBIENTE_VIRTUAL>
     # Mostrar lo paquetes instalados en un ambiente virtual
     $ workon <NOMBRE_DEL_AMBIENTE_VIRTUAL>
     $ pip freeze
```

6. Instalar Django 1.4.5

(hay que asegurarnos de que tenemos activado el ambiente virtual en donde lo queremos instalar)

```
# instalar django 1.4.5
$ pip install Django== 1.4.5


# Instalar docutils, utilizado para el administrador de #Django
$ pip install docutils

# Si se quiere probar Django, hacer un proyecto de prueba
$ django-admin.py startproject <NOMBRE_DE_LA_APP>
$ cd <NOMBRE_DE_LA_APP>
# Darle permisos de ejecución a manage.py
$ chmod +x manage.py
# correr el servidor
$ ./manage.py runserver 0.0.0.0:8000
```

7. Instalar la librería de imágenes Pillow

```
# Instalar librerías
$ sudo apt-get install libjpeg8-dev libfreetype6-dev zlib1g-dev

# Instalar pillow
$ pip install pillow
```

8. Instalar MySQL

```
$ sudo apt-get install mysql-server libmysqlclient-dev
$ pip install MySQL-Python
```

9. Instalar South

South nos ayuda con las migraciones y los cambios de esquema dentro de la base de datos

```
# Instalar south
$ pip install south


# Agregar South a nuestras INSTALLED_APPS (en settings.py)
$ vim <TU_APP>/settings.py
INSTALLED_APPS =
    ...
    'south',
    ...
```

10. Instalar memcached

Memcached incrementa el rendimiento y minimiza las consultas a tu base de datos con ayuda de un key:value que se almacena en memoria y que es bastante rápido y sencillo.

```
# Instalar el librerías y servidor con memcach
$ sudo apt-get install memcached libmemcached-dev

# Instalar pylibmc
$ pip install pylibmc

# Editar en settings.py los ajustes de CACHES
$ vim <TU_APP>/settings.py
CACHES = {
    'default': {
        'BACKEND':
'django.core.cache.backends.memcached.PyLibMCCache',
        'LOCATION': '127.0.0.1:11211',
    }
}
```

11. Instalar RabbitMQ

Para realizar la ejecución asíncrona de tareas se requiere de un servidor que administre las tareas que los diferentes trabajadores de celery (paquetería utilizadoapara la ejecución asíncrona de tareas) van a estar realizando. Para esto utilizamos RabbitMQ

```
$ sudo apt-get install rabbitmq-server
$ sudo rabbitmqctl add_user <USUARIO_RABBIT> <CONTRASEÑA>
$ sudo rabbitmqctl add_vhost <RABBIT_VHOST>
$ sudo rabbitmqctl set_permissions -p <RABBIT_VHOST>
<USUARIO_RABBIT> ".*" ".*" ".*"
```

12. Instalar Celery

Celery es la paquetería que nos permite ejecutar tareas asíncronas, para poderlos instalar esta paquetería se deben configurar algunas cosas.

```
# Instalar celery
```

```
$ pip install django-celery

# Editar algunas configuraciones en settings.py
$ vim <TU_APP>/settings.py

# Agregar djcelery a  INSTALLED_APPS
INSTALLED_APPS =
    ...
    'djcelery',
    ...

# Agregar estas configuraciones(seguimos settings.py)
BROKER_URL = "amqp://
<USUARIO_RABBIT>:<CONRASEÑA_RABBIT>@localhost:5672/<RABBIT_VHOST>"
CELERY_RESULT_BACKEND = "database"

# Elegir la configuración que concuerde con la configuración de tu
base de datos
CELERY_RESULT_DBURI = "mysql://<USUARIO_DE_LA_BD>:
<CONTRASEÑA_DE_LA_BD>@localhost/<NOMBRE_DE_LA_BD>"
CELERY_RESULT_DBURI = "postgresql://<USUARIO_DE_LA_BD>:
<CONTRASEÑA_DE_LA_BD>@localhost/<NOMBRE_DE_LA_BD>"

# Poner al final del settings.py las siguientes lineas
import djcelery
djcelery.setup_loader()


# Comandos útiles para celeryd
     # Iniciar celeryd con Beat and Event
     $ ./manage.py celeryd -B -E

     # Iniciar celeryd
     $ ./manage.py celeryd
```

13. Instalar Gunicorn

Gunicorn es el WSGI (web server gateway interface) que se va a utilizar. Un WSGI es la manera en la que nuestra aplicación se va a comunicar con el servidor web.

```
# Instalar gunicorn
$ pip install gunicorn

# Agregar gunicorn dentro de INSTALLED_APPS(settings.py)
$ vim <YOUR_APP>/settings.py
INSTALLED_APPS =
    ...
    'gunicorn',
    ...
# Probar gunicorn
$ ./manage.py run_gunicorn -w 4 -k gevent
```

14. Instalar Supervisor

Supervisor va a administrar los servicios de Celery, Celerycam, y Gunicorn. Para esto se deben de escribir los archivos de configuración celeryd.conf, celerycam.conf y gunicorn.conf.

```
# Instalar supervisor
$ sudo apt-get install supervisor

# Para la configuración de celeryd
# En /etc/supervisor/conf.d/celeryd.conf escribir
# Nombre del servicio
[program:celeryd]


# Comando para iniciar celery
```

```
command = /home/<USERNAME>/.virtualenvs/<NOMBRE_DE_VIRTUALENV>/
bin/python /home/<USERNAME>/<NOMBRE_DE_LA_APP>/manage.py celeryd -
B -E

# TIP: Para probar que la ruta sea la correcta puedes probar #el
comando directamente en consola

# Directorio en el que vamos a estar al correr command
directory = /home/<USERNAME>/<NOMBRE_DE_LA_APP>

# Nombre del usuario con el que vamos a correr el comando
user = <USERNAME>

# Correr el comando cuando se inicie o reiniciar si falla
autostart = true
autorestart = true

# Quitar stdout y stderr de celery y escribir estos archivos de
log
stdout_logfile = /var/log/supervisor/celeryd.log
stderr_logfile = /var/log/supervisor/celeryd_err.log



# Para la configuración de celerycam
# En /etc/supervisor/conf.d/celerycam.conf escribir
# Nombre del programa

[program:celerycam]
command = /home/<USERNAME>/.virtualenvs/<NOMBRE_DEL_VIRTUALENV>/
bin/python /home/<USERNAME>/<NOMBRE_DE_LA_APP>/manage.py celerycam
directory = /home/<USERNAME>/<NOMBRE_DE_LA_APP>
user = <USERNAME>
autostart = true
autorestart = true
stdout_logfile = /var/log/supervisor/celerycam.log
stderr_logfile = /var/log/supervisor/celerycam_err.log
# Para la configuración de gunicorn
# En /etc/supervisor/conf.d/gunicorn.conf escribir
```

```
# Nombre del programa
[program:gunicorn]
command = /home/<USERNAME>/.virtualenvs/<NOMBRE_DEL_VIRTUALENV>/
bin/python /home/<USERNAME>/<NOMBRE_DE_LA_APP>/manage.py
run_gunicorn -w 4 -k gevent
directory = /home/<USERNAME>/<NOMBRE_DE_LA_APP>
user = <USERNAME>
autostart = true
autorestart = true
stdout_logfile = /var/log/supervisor/gunicorn.log
stderr_logfile = /var/log/supervisor/gunicorn_err.log
```

15. Reiniciar Supervisor

```
$ sudo service supervisor restart

# Comandos útiles de supervisor
     # Reiniciar/detener/comenzar todos los servicios
     #administrados por supervisor
     $ sudo supervisorctl restart all
     $ sudo supervisorctl stop all
     $ sudo supervisorctl start all

     # Reiniciar únicamente celeryd
     $ sudo supervisorctl restart celeryd

     # Comenzar únicamente gunicorn
     $ sudo supervisorctl start gunicorn
```

16. Instalar Nginx

```
# Instalar nginx
$ sudo apt-get install nginx
```

```
# Crear y editar los archivos de configuración para nginx
# Remover el enlace simbólico que está por default
$ sudo rm /etc/nginx/sites-enabled/default

# Crear un nuevo archivo de configuración y hacerle un enlace
#simbólico
$ sudo touch /etc/nginx/sites-available/<TU_APP>
$ cd /etc/nginx/sites-enabled
$ sudo ln -s ../sites-available/<TU_APP>

# Editar el archivo de configuración
$ vim /etc/nginx/sites-available/<TU_APP>
# Así se debe ver el archivo
# Definir un servidor upstream llamado gunicorn en el puerto 8000
upstream gunicorn {
    server localhost:8000;
}

# Hacer un servido nginx
server {
    # Escuchar en el puerto 80
    listen 80;

    # Para solicitudes a estos dominios
    server_name <TU_DOMINIO>.com www.<TU_DOMINIO>.com;

    # Buscar en este directorio los archivos a servir
    root /var/www/;

    # Guardar los logs en los siguientes archivos
    access_log /var/log/nginx/<TU_APP>.access.log;
    error_log /var/log/nginx/<TU_APP>.error.log;

    client_max_body_size 0;

    try_files $uri @gunicorn;
```

```
    location @gunicorn {

        client_max_body_size 0;


        proxy_pass http://gunicorn;


        proxy_redirect off;

        # Definir el tiempo de respuesta necesario
        proxy_read_timeout 5m;

        proxy_set_header Host            $host;
        proxy_set_header X-Real-IP       $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
    }
}
```

Referencias

Apreche.net (2012) “Complete Single Server Django Stack Tutorial”. Rescatado de http://www.apreche.net/complete-single-server-django-stack-tutorial/ el 27 de Marzo de 2013.

Ligas de interés

http://www.mysql.com/

http://south.aeracode.org/

http://celeryproject.org/

http://www.rabbitmq.com/

http://gunicorn.org/

http://supervisord.org/

http://nginx.com/

Contacto

Arturo Jamaica García
arturo@brounie.com

Jorge Castillo Ocadiz
jorge@brounie.com

www.brounie.com