

# ¿Quién soy?

Mi nombre debe salir en el programa

Con "estudios" en Ciencias de la Computación

Más de 8 años trabajando como sysadmin / devops / SRE / <poner etiqueta tonta aquí>

Más info en http://bit.ly/1sT7bLw

Dejar en la puerta todas las expectativas que tengas, por favor



## Checklist

¿Qué es Fabric?
Primeros pasos
Subiendo el nivel
Usar Fabric como biblioteca
(Gracias Andrey)



## ¿Qué es Fabric?

Fabric es una biblioteca de Python (2.5-2.7) y herramienta de línea de comandos para facilitar el uso de SSH para el despliegue de aplicaciones o tareas de administración de sistemas. Ya sabes de forma pythonica La estructura de algo / Tela / Un club nocturno en Londres / Un proxy para toysql http://www.fabfile.org/en/latest/



# ¿Qué es Fabric?

# Y python3?





## ¿Qué es Fabric?

¡Calma! Se está cocinando Fabric 2 y bibliotecas como invoke

http://www.fabfile.org/en/latest/roadmap.html #invoke-fabric-2-x-and-patchwork

http://www.pyinvoke.org/

Todo esto está en desarrollo y como sabrán a los sysadmin se nos juzga por la estabilidad de los sistemas



## Primeros pasos

```
Instalar
 pip install fabric
Crear un fabfile.py
from fabric.api import run
def host_type():
    run('uname -s')
```



## Primeros pasos

```
Ejecutar
fab -H host host_type
[host] run: uname -s
[host] out: Linux
```



## Primeros pasos

```
Un poco más del API de Fabric
 local
 sudo
 settings
 Cd
 env.hosts
```







Usando Fabric como Func / Mcollective / Ansible command

```
fab -H web01, web02 -- "cd /var/www && pwd"
```

```
[web01] Executing task '<remainder>'
```

[web01] run: cd /var/www && pwd

[web01] out: /var/www

[web01] out:



Usando Fabric como Func / Mcollective / Ansible command

```
[web02] Executing task '<remainder>'
```

[web02] run: cd /var/www && pwd

[web02] out: /var/www

[web02] out:

Done.



```
Usando el decorador task
 Antes
from fabric.api import run
def private():
    run('pwd')
def public():
    # preparar cosas
    private()
```



Usando el decorador task
Antes
fab -1
Available commands:

private
public



```
Usando el decorador task
 Después
from fabric.api import run, task
def private():
    run('pwd')
@task
def public():
    # preparar cosas
    private()
```



Usando el decorador task Después fab -1

Available commands:

public



#### Documentando las funciones

```
from fabric.api import task
```

```
@task
def deploy_stuff(version):
    '''Deploy the specified version of stuff
    Keyword arguments:
    version -- version to deploy
```



#### Documentando las funciones

Example:

```
$ fab deploy:release_X.Y.Z
```

1 1 1



Documentando las funciones fab -1

Available commands:

deploy\_stuff Deploy the specified version of stuff



#### Documentando las funciones

```
fab -d deploy_stuff
Displaying detailed information for
task 'deploy_stuff':
```

Deploy the specified version of stuff



#### Documentando las funciones

**Keyword arguments:** 

version -- version to deploy

Example:

\$ fab deploy:release\_X.Y.Z



#### Decorador execute

```
from fabric.api import cd, execute, sudo, task
```



#### Decorador execute

```
@task
def deploy_stuff(version):
    '''Deploy the specified version
of stuff
...
```



#### Decorador execute



#### Decorador execute

Para que execute funcione correctamente las tareas llamadas por execute no deben tener hosts asociados, sino al llamar la tarea se ejecutara múltiples veces, se puede evitar usando el decorador runs\_once



```
Decorador execute
...
env.use_ssh_config = True
@task
@hosts('web01', 'web02')
def alpha():
```

execute(beta)



```
Decorador execute
@hosts('web03', 'web04')
def beta():
    execute(gamma)
def gamma():
    run('hostname')
```



#### Decorador execute

```
fab alpha
```

```
[web01] Executing task 'alpha'
```

```
[web03] Executing task 'beta'
```

```
[web03] run: hostname
```

[web03] out: web03

[web03] out:



#### Decorador execute

```
[web04] Executing task 'beta'
```

[web04] run: hostname

[web04] out: web04

[web04] out:



#### Decorador execute

```
[web02] Executing task 'alpha'
```

[web03] Executing task 'beta'

[web03] run: hostname

[web03] out: web03

[web03] out:



#### Decorador execute

```
[web04] Executing task 'beta'
```

```
[web04] run: hostname
```

[web04] out: web04

[web04] out:

Done.



Separar en "módulos" directorio

fabfile.py

proyecto01.py

proyecto02.py



Separar en "módulos" fabfile.py

```
from fabric.api import env
import proyecto01
import proyecto02
```

env.use\_ssh\_config = True



```
Separar en "módulos"
proyecto01.py
from fabric.api import execute, task
@task
def deploy(version):
    '''Deploy the specified version of stuff
    1 1 1
```



```
Separar en "módulos"
proyecto02.py
from fabric.api import execute, task
@task
def shutdown():
    '''Shutdowns machine'''
```



Separar en "módulos"

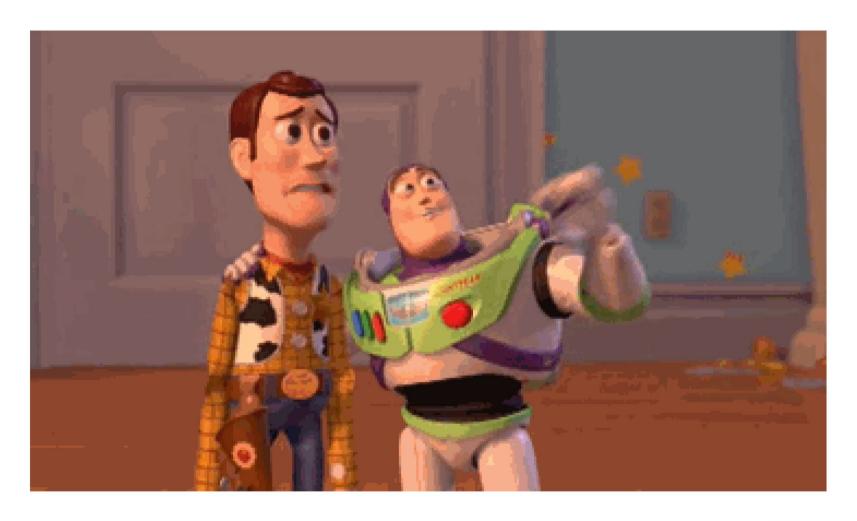
fab -1

Available commands:

proyecto01.deploy\_stuff Deploy the
specified version of stuff
proyecto02.shutdown Shutdown machine

fab proyect01.deploy\_stuff:release\_X.Y.Z







#### Ejemplo usando boto

```
from fabric.api import env, execute, run,
task
import boto.ec2
```

```
REGION = 'eu-west-1'
```

```
env.skip_bad_hosts = True
env.use_ssh_config = True
```



#### Ejemplo usando boto

```
def get_hosts_list_by_role(role):
    conn = boto.ec2.connect_to_region(REGION)
    filters = {
        'instance-state-name': 'running',
        'tag:role': role,
    return [instance.private_ip_address
            for instance in
conn.get_only_instances(filters=filters)]
```



```
Ejemplo usando boto
def kick_it(what):
    run('restart {what}'.format(what=what))
@task
def restart_faulty_workers():
    worker_hosts =
get_hosts_list_by_role(role='worker')
    if worker hosts:
        execute(kick_it, hosts=worker_hosts,
what='pehachepe')
```



## Usar Fabric como biblioteca

```
from fabric.api import env, run
from fabric.network import disconnect_all
def check_log(jobid):
    env.use_ssh_config = True
    env.host_string = 'hostname'
    try:
        run('LANG="C" grep -C 10 {jobid}
log'.format(jobid=jobid))
    finally:
        disconnect_all()
```



## Usar Fabric como biblioteca

```
from fabric.api import env, execute, run
from fabric.network import disconnect_all
def func():
    env.use_ssh_config = True
    env.warn_only = True
    try:
        f = run('ls -lR /', pty=False, combine_stderr=False)
        print('Return code is: {}'.format(f.return_code))
        print('STDERR has:')
        print(f.stderr)
    finally:
        disconnect_all()
```

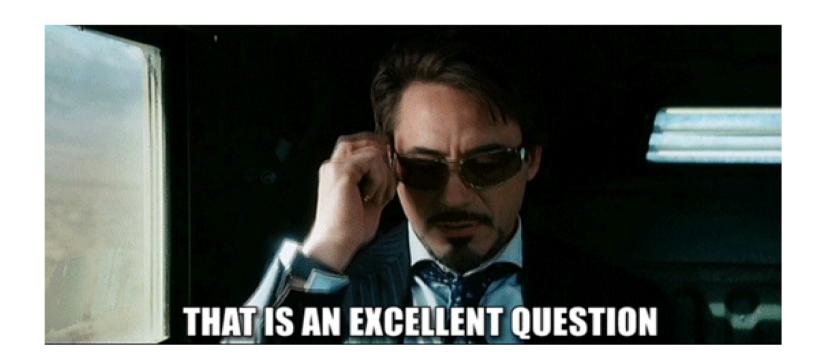


#### Usar Fabric como biblioteca

```
def run_stuff():
    execute(func, host='hostname')
def main():
    run_stuff()
if ___name__ == '___main___':
    main()
```



## ¿Preguntas?





## Un poco de viralidad





## Gracias





## Créditos

http://upload.wikimedia.org/wikipedia/commons/c/c1/Basilica\_del\_Pilarsunset.jpg

http://upload.wikimedia.org/wikipedia/commons/4/4a/Python3-powered\_hello-world.svg

http://i.giphy.com/JtwZyPicr2vTi.gif

http://www.mariowiki.com/images/thumb/6/68/SMB\_W1-1 Glitch3.gif/180px-SMB W1-1 Glitch3.gif

http://static.ow.ly/photos/original/7uAzQ.png

http://i.giphy.com/140ObFj9MRjRIc.gif

http://i.giphy.com/dXICCcws9oxxK.gif

http://i.giphy.com/1Z02vuppxP1Pa.gif

