



LO BÁSICO COMPLEMENTA LO COMPLEJO: VIRTUALENV PARA NOVATOS

Abdel G. Martínez L.

AGENDA

Lo básico complementa lo complejo: virtualenv para novatos

- ⦿ Pequeños pasos iniciales
- ⦿ Típico problema de pitón
- ⦿ Dando un grano de arena
- ⦿ Lo que es y lo que no es
- ⦿ Representación gráfica
- ⦿ Distintos puntos de vista
- ⦿ Entonces, ¿por qué?
- ⦿ Creando la necesidad
- ⦿ Contrastando alternativas
- ⦿ Existe algo mucho mejor
- ⦿ Demostración

PEQUEÑOS PASOS INICIALES

- ⦿ pip es una herramienta para instalar y administrar paquetes propios de Python.
 - ⦿ Es un reemplazo para easy_install
- ⦿ PyPi (Python Package Index) es un repositorio de software para Python con más de 60000 paquetes.
 - ⦿ <https://pypi.python.org/pypi>
- ⦿ Sin embargo, esto no es suficiente...

TÍPICO PROBLEMA DE PYTHON

- ◎ ¿Cómo instalo paquetes?
- ◎ ¿Cómo utilizo diferentes versiones de Python?
- ◎ ¿Cómo instalo diferentes versiones de paquetes?
- ◎ ¿Cómo pruebo paquetes sin dañar mi sistema?
- ◎ ¿Cómo entrego código de Python a otros?
- ◎ ¿Cómo lo verifico?

DANDO UN GRANO DE ARENA

- ⦿ Fue escrito por Ian Bicking.
- ⦿ Es patrocinado por Open Planning Project.
- ⦿ El código está disponible bajo MIT en GitHub.
- ⦿ Mantenido por un grupo de desarrolladores.
- ⦿ Las fechas de lanzamiento están ligadas a las de pip.



LO QUE ES Y LO QUE NO ES

¿Qué es?

- Herramienta de virtualización del entorno Python.
- No representa virtualización completa.
- Separa el fichero ejecutable, junto a un conjunto de librerías.

¿Qué no es?

- Simulación del entorno en producción.
- No contempla aplicaciones del entorno.

REPRESENTACIÓN GRÁFICA



DISTINTOS PUNTOS DE VISTA

Ventajas

- ⦿ Manejo de versiones de paquetes instalados, por proyecto.
- ⦿ No hay riesgo de afectar los paquetes del proyecto.

Desventajas

- ⦿ Demasiado detalle, seguir fallas de seguridad y 'bugs'.
- ⦿ No es simple, visibilidad demasiado amplia de los paquetes.
- ⦿ Problemas de integración con entornos heterogéneos.

ENTONCES, ¿POR QUÉ?



Aislamiento



Permisos



Organización



No-Globalización

CREANDO LA NECESIDAD

INSTALACIÓN – GESTOR DE PAQUETES

Fedora

```
$ sudo yum install python-virtualenv
```

Ubuntu

```
$ sudo apt-get install python-virtualenv
```

CREANDO LA NECESIDAD

INSTALACIÓN – GESTOR PIP

Linux, Mac OS X

```
$ sudo pip install virtualenv
```

Fedora

```
$ sudo pip-python install virtualenv
```

CREANDO LA NECESIDAD

CREACIÓN DE ENTORNO VIRTUAL

```
$ virtualenv nombre_proyecto
```

nombre_proyecto/

bin/

include/

lib/

CREANDO LA NECESIDAD

ACTIVACIÓN DE ENTORNO VIRTUAL

```
$ cd nombre_proyecto  
$ source bin/activate  
(nombre_proyecto)$
```

DESACTIVACIÓN DE ENTORNO VIRTUAL

```
(nombre_proyecto)$ deactivate  
$
```

CREANDO LA NECESIDAD

INSTALACIÓN DE PAQUETES EN ENTORNO

```
(nombre_proyecto)$ pip install pisa
```

CONTRASTANDO ALTERNATIVAS

workingenv

- Predecesor, utiliza el intérprete
- Se basa en \$PYTHONPATH

virtual-python

- Predecesor, utiliza enlaces simbólicos
- Apunta toda la librería estándar

zc.buildout

- No crea ambiente aislado
- Utiliza archivo de configuración

EXISTE ALGO MUCHO MEJOR

- ⦿ Doug Hellman liberó `virtualenvwrapper`.
- ⦿ Extensiones que incluyen wrappers para mejorar el flujo de trabajo.
- ⦿ Organiza los entornos virtuales en un solo lugar.
- ⦿ Manejo avanzado de entornos.
- ⦿ Totalmente extensible.



¿PREGUNTAS?

Abdel G. Martínez L.

@abdelgmartinezl

<http://abdelmartinez.com>