PYTHONEAR

SIN

SALPICAR

Ande ereh, ande venihte, ande váh

- Cristóbal Contreras Rubio
- Calvo
- Flamer ecléctico

- Multipotencialista (postureo)
- Un sæcio más de Almería

- Estudié la carrera en ***Hacklab Almería***
- Mi TFG fue "***The Pymiento Project***"
- Becario en la startup ***La Jaquería***
- Estropeo cosas con Python

PYTHONEAR SIN ~~GUARREAR~~

PYTHONEAR SIN ~~ENFANGAR~~

PYTHONEAR SIN SALPICAR

El objetivo es pasar de esto...

A esto...

¿POR QUÉ PYTHON?

1. La comunidad:
   - Gente muy maja
   - Inclusiva
   - Feminista (todos somos iguales)

2. ZEN:
   - Filosofía de la simplicidad y la belleza
   - Cuanto más fácil y sencillo, más pro.

3. Buena y extensa documentación:
   - El lenguaje tiene documentación oficial
   - Sus librerías
   - Tutoriales

4. Guía de estilos:
   - PEP8
   - Los demás PEP

5. Gestor de paquetes:
   - pip

6.Multipropósito y multiparadigma:
   - Para hacer webs (backend)
   - Programas de escritorio
   - Automatizar tareas
   - Apps móviles
   - ...

Empezando con ~~Phyton~~ Python 1.0

* La ~~librería~~ biblioteca standar tiene de todo (viene completica).
* Hay más ~~librerías~~ bibliotecas chachis de terceros (requests, pillow, ...).
* Contamos con un gestor de paquetes que nos soluciona la vida, PIP.
* PIP es droga en vena, cuando lo pruebas...

¡¡¡NO PARAS!!!

Empezando con ~~Phyton~~ Python 2.0

* Se te empieza a ir la olla con pip instalando cosas.

* E
ver

* L
qui

"Amigos y amigas, bienvenidos y bienvenidas,
un día más, a un nuevo programa ~~de Bricomanía~~"

ENTORNOS VIRTUALES

* Python cuenta con una herramienta para cacharrear llamada "Entornos virtuales".
* No, no es una máquina virtual (ni contenedores).

* Los entornos virtuales son instalaciones aisladas de Python (una carpetica con tu Python, tu pip y tus paqueticos instalados).
* La idea de los entornos virtuales es que puedas trabajar de modo independiente en cualquier sistema.
* Existen unos ficheros, **requirements.txt**, que junto con esto, te permiten poder tener tu entorno de desarrollo en segundos.

virtualenv

Documentasión ofishiah → **Pínchame**

Instalació →

```
# Linux - MacOS - Windows
pip install virtualenv
```

Desde la versión 3.3 existe un mini versión de virtualenv en la biblioteca standard llamada venv → **Pínchame (si quieres bichearla)**

virtualenv

```
# Crear un virtualenv
# virtualenv <nombre> [--python=<ruta al python elegido>]
virtualenv env --python=python3
# Activar el virtualenv en Linux-MacOS
source /../env/bin/activate
# Activar el virtualenv en Windows
\..\env\Scripts\activate
# Desactivar el virtualenv
deactivate
# Borrar el virtualenv = borrar su carpeta
```

```
env/
  bin/
  include/
  lib/
    site-packages/
```

virtualenvwrapper

Documentasión ofishiah → **Pínchame**

Instalació →

```
# Linux - MacOS - Windows
pip install virtualenvwrapper
```

```
# Windows
pip install virtualenvwrapper-win
```

Al instalarse genera un script - **virtualenvwrapper.sh**

Este script debe ser cargado por la terminal al arrancar, así que debemos averiguar su ubicación para ser configurado

```
# Linux - MacOS
sudo find / -name virtualenvwrapper.sh -type f
```

virtualenvwrapper

Configuración en el script de tu shell (.zshrc / .bshrc / ...)

1) ***VIRTUALENVWRAPPER_PYTHON***

   Interprete de Python con virtualenvwrapper

2) ***WORKON_HOME***

   Donde vas a tener todos los virtualenvs

3) ***PROJECT_HOME***

   Carpeta donde tienes todos los programas que haces

4) Donde está el virtualenvwrapper.sh

Al final de tu fichero .zshrc / .bshrc debes de tener algo así

```
# Virtualenvwrapper
VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3.6
export WORKON_HOME=$HOME/.local/share/virtualenvs
export PROJECT_HOME=$HOME/Pogramas
source /usr/local/bin/virtualenvwrapper.sh
```

virtualenvwrapper

```
# Crear un virtualenv
# mkvirtualenv <nombre> [--python=<interprete python>]
mkvirtualenv env --python=python3
# Ver que virtualenvs hay
```

```
workon
# Activar un virtualenv
workon env
# Desactivar el virtualenv
deactivate
# Borrar el virtualenv
rmvirtualenv
```

Pipenv

Documentasión ofishiah → **Pínchame**

Instalació →

```
# Linux - MacOS - Windows
pip install pipenv
```

* Es otra genialidad de Kenneth Reith.
* Combina pip + virtualenv.
* Añade seguridad (un ***requirements.txt*** extendido).

  Pipfile + ***Pipfile.lock***
* Sirve también para empaquetado.
* Se va a acabar convirtiendo en el standar

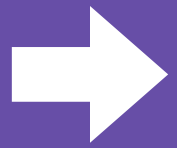
Pipenv

```
# Crear un virtualenv
# pipenv [--python <interprete python>]
pipenv --python python3.6
# Instalar dependencias del Pipfile
pipenv install

# Instalar dependencias
pipenv install requests
# Entrar al virtualenv
pipenv shell
# Ejecutar un comando
run python main.py
# Borrar el virtualenv
pipenv -rm
```

pyenv

Documentasión ofishiah



Pínchame
(mirar su web para la instalación)

* No está hecho en Python.
* Solo está disponible en Linux-MacOS.

  Hay un fork para Windows (**píncha aquí**).

* Permite tener distintos interpretes de Python en el ordenador.
* No solo puedes tener interpretes de CPython (C), puedes tener todos los demás: PyPy (Python), Jython (Java), IronPython (C#), MicroPython ...

pyenv

```
# Instalar un interprete de Python
pyenv install 3.7.1
# Ver interpretes instalados
pyenv versions
```

```
# Cambiar la versión global de Python (no recomendado)
pyenv global 3.7.1
# Deshacer lo de antes, ya que no debes hacerlo
pyenv global system
# Definir un interprete para la carpeta donde estás
pyenv local 3.7.1
# Desinstalar un interprete de Python
pyenv uninstall 3.7.1
```

Recomendaciones

Pipenv + **pyenv**

Usa Pipenv si o si

pyenv te va a facilitar a vida

virtualenvwrapper te va a servir para saber que virtualenvs tienes desperdigados y poder borrarlos

muNchas

graciah

sæcios