

CREACIÓN DE UN PROYECTO DJANGO

EXERCISES QUE TRABAJAREMOS EN EL CUE

O

- EXERCISE 1: VERIFICACIÓN E INSTALACIÓN DE PYTHON 3.
- EXERCISE 2: VERIFICACIÓN DE LA INSTALACIÓN DE PAQUETE PIP 3 PYTHON PACKAGE INDEX (PYPI).
- EXERCISE 3: INSTALACIÓN DEL ENTORNO VIRTUAL (VIRTUALENVWRAPPER).
- EXERCISE 4: INSTALACIÓN DE DJANGO EN EL ENTORNO VIRTUAL.
- EXERCISE 5: INSTALACIÓN DE PYTHON Y EL ENTORNO VIRTUAL DE DJANGO EN WINDOWS 10.

EXERCISE 1: VERIFICACIÓN E INSTALACIÓN DE PYTHON 3.

El objetivo del presente ejercicio es plantear una guía paso a paso para la virtualización de un proyecto de desarrollo en Django.

Requisitos previos:

- Tener conocimiento de una terminal o línea de comando, e instalación de paquetes de software en el sistema operativo, en este caso Linux Ubuntu y Windows 10.
- Tener previamente instalada la versión de Python 3.

PROCEDIMIENTOS DE LA PRÁCTICA

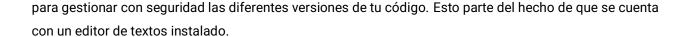
El entorno de desarrollo de Django es una instalación en el computador local que se utiliza para el desarrollo de aplicaciones, con la finalidad de probarlas antes de desplegarlas al entorno de producción.

Django proporciona herramientas, y también un conjunto de scripts de Python que permiten crear y desarrollar proyectos Django, basados en un servidor web de desarrollo simple, el cual se usa para probar de forma local las aplicaciones web Django con un navegador web del computador.

Existen un conjunto de herramientas periféricas, que forman parte del entorno de desarrollo, como lo es un editor de textos o IDE para editar código (VSC), una herramienta de gestión del control de fuentes como Git



CREACIÓN DE UN PROYECTO DJANGO



El framework Django se ejecuta por encima de Python, y puede utilizarse con Python 2, o con Python 3 (o ambos, inclusive). Para ello se debe tener en cuenta que:

- Python 2 es una versión tradicional del lenguaje, la cual no va a tener más características nuevas, pero que tiene disponible para los desarrolladores un enorme repositorio de bibliotecas de terceros de alta calidad (algunas de las cuales no están disponibles en Python 3).
- Python 3 es una actualización de Python 2 que, aunque similar, es más consistente y fácil de usar pues se encuentra en constante mejoramiento y evolución.

VERIFICACIÓN E INSTALACIÓN DE PYTHON 3

0

El framework Django tiene como requerimiento que se debe tener instalado Python en el sistema operativo. Si se tiene instalado Python 3, se necesita la Python Package Index — pip3 —, que se usa para gestionar (instalar, actualizar y eliminar) los paquetes/bibliotecas Python utilizados por Django y otras aplicaciones en Python.

Para verificar si tenemos instalado Python, procedemos a abrir una consola terminal en Linux, y escribimos:

```
1 $ python -V
2 Python 3.8.3
```

Observamos que tenemos instalado la versión de Python 3.8.3.

Si no tenemos instalado Python, procedemos a instalar la última versión.

Primero, instale el requisito previo para agregar PPA personalizados:

```
1 $ sudo apt install software-properties-common -y
```

En segundo lugar, agregue el deadsnakes/ppa a la lista de fuentes del administrador de paquetes de APT:



CREACIÓN DE UN PROYECTO DJANGO

1 \$ sudo add-apt-repository ppa:deadsnakes/ppa -y

Se actualiza el listado de paquetes disponibles:

0

1 \$ sudo aptitude update

Instalación de Python:

1 \$ sudo aptitude install python3

Verificación de la instalación:

1 \$ python -V 2 Python 3.8.3

EXERCISE 2: VERIFICACIÓN DE LA INSTALACIÓN DE PAQUETE PIP 3 - PYTHON PACKAGE INDEX (PYPI).

Ubuntu Linux incluye Python 3 por defecto. Sin embargo, la herramienta Python Package Index que se requiere para instalar paquetes de Python 3, como por ejemplo Django, no está disponible por defecto. Se instala de la siguiente manera en la consola:

1 \$ sudo aptitude install python3-pip3

Para verificar que se encuentra instalado, podemos revisar listando los paquetes instalados:

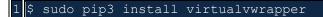
1 \$ pip3 list

EXERCISE 3: INSTALACIÓN DEL ENTORNO VIRTUAL (VIRTUALENVWRAPPER).

Se utilizarán las bibliotecas virtualenvwrapper en Linux para crear los entornos virtuales (disponible para Mac OS y Windows), que utiliza como base la herramienta virtualenv. Esta herramienta wrapper crea una interfaz consistente para la gestión de interfaces en todas las plataformas. Para instalar virtualenvwrapper, se realizará por medio de pip3:



• CREACIÓN DE UN PROYECTO DJANGO



0

Posteriormente, se debe agregar al archivo de inicio de la terminal o shell oculto **.bashrc**, que se encuentra en el directorio home del usuario, las siguientes líneas al final:

```
1 $ nano .bashrc
```

```
##### Entornos Virtuales Python ############

export WORKON_HOME=$HOME/.virtualenvs

export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3

export PROJECT_HOME=$HOME/Dev-Python

source /usr/local/bin/virtualenvwrapper.sh
```

Las líneas anteriores permiten ajustar la localización de donde deberían estar alojados los entornos virtuales, la de los directorios de tus proyectos de desarrollo, y la del script instalado con este paquete.

Se recarga el archivo de inicio, o se reinicia la terminal:

```
1 $ source ~/.bashrc
```

Una vez que hayas instalado **virtualenvwrapper**, se procede a crear un nuevo entorno virtual con el comando **mkvirtualenv**.

```
1 $ mkvirtualenv django_env
```

Al finalizar, observamos que el nuevo entorno virtual se encuentra activo, y que el comienzo del prompt será el nombre del entorno entre paréntesis.



0

CREACIÓN DE UN PROYECTO DJANGO

```
11 virtualenvwrapper.user scripts
                                                                      creating
12 /home/luispc/.virtualenvs/django env/bin/predeactivate
13 virtualenvwrapper.user scripts
                                                                      creating
14 /home/luispc/.virtualenvs/django env/bin/postdeactivate
15 virtualenvwrapper.user_scripts
                                                                      creating
16 /home/luispc/.virtualenvs/django_env/bin/preactivate
17 virtualenvwrapper.user scripts
                                                                      creating
18 /home/luispc/.virtualenvs/django env/bin/postactivate
19 virtualenvwrapper.user scripts
                                                                      creating
20 /home/luispc/.virtualenvs/django env/bin/get env details
  (django env) (base) dev-django@FullStack-Dev:~$
```

Para el uso del entorno virtual, tenemos los siguientes comandos más usados:

deactivate — Salir del entorno virtual Python actual.

```
1 $ (django_env) (base) dev-django@FullStack-Dev:~$ deactivate
```

workon — Listar los entornos virtuales disponibles.

```
1 $ (django_env) (base) dev-django@FullStack-Dev:~$ workon 2 django_env 3 my_django_environment
```

• workon nombre_entorno_virtual — Activar el entorno virtual Python especificado.

```
1 $ django_env base) dev-django@FullStack-Dev:~$ workon django_env 2 (django_env) (base) dev-django@FullStack-Dev:~$
```

rmvirtualenv nombre_entorno_virtual — Borrar el entorno especificado.

```
1 $ django_env)base)dev-django@FullStack-Dev:~$ rmvirtualenv django_env 2 Removing django_env...
```

EXERCISE 4: INSTALACIÓN DE DJANGO EN EL ENTORNO VIRTUAL.

Iniciamos, y entramos al ambiente virtual django_env, donde procedemos a instalar Django con pip3:



CREACIÓN DE UN PROYECTO DJANGO

```
1 $ (base) dev-django@FullStack-Dev:~$ workon django_env 2 (django_env) (base) luispc@FullStack-Dev:~$ pip3 install django
```

Verificamos los paquetes instalados en el entorno virtual con pip list:

0

Observamos que tenemos instalado Django en la versión 4.0.5.

EXERCISE 5: INSTALACIÓN DE PYTHON Y EL ENTORNO VIRTUAL DE DJANGO EN WINDOWS 10.

El sistema operativo Windows no incluye por defecto Python. En este caso, se debe instalar siguiendo el procedimiento como con cualquier paquete en Windows, junto con la herramienta pip3 desde python.org.

- 1. Procedemos a descargar el instalador requerido, para ello vamos a: https://www.python.org/downloads/
- 2. Selecciona el botón de Descarga python-3.9.13-amd64.
- 3. Se inicia el proceso de descarga.
- 4. Procedemos a instalar Python haciendo doble-clic al archivo descargado, y pulsando siguiente en las ventanas de instalación. Aquí se selecciona que se agregue automáticamente la ruta path al sistema.
- 5. Finalmente, verificamos que se encuentra instalado abriendo la consola de comandos, y colocando lo siguiente:



0

CREACIÓN DE UN PROYECTO DJANGO

```
Símbolo del siste
C:\Users\luispc>python -V
Python 3.9.13
C:\Users\luispc>pip -V
pip 22.0.4 from C:\Users\luispc\AppData\Local\Programs\Python\Python39\lib\site-packages\pip (py
thon 3.9)
C:\Users\luispc>pip list
Package
                      Version
distlib
                      0.3.5
filelock
                      3.7.1
pip
                      22.0.4
platformdirs
                      2.5.2
setuptools
                      58.1.0
/irtualenv
                       20.16.2
virtualenvwrapper-win 1.2.7
 IARNING: You are using pip version 22.0.4; however, version 22.2.1 is available.
 ou should consider upgrading via the 'C:\Users\luispc\AppData\Local\Programs\Python\Python39\py
 :\Users\luispc>
```

INSTALACIÓN DEL ENTORNO VIRTUAL EN WINDOWS 10

Instalar virtualenvwrapper-win es incluso más simple que poner en marcha virtualenvwrapper, pues no necesitas configurar donde se almacena la herramienta la información del entorno, sino que el mismo se encuentra por defecto. Lo que se necesita es ejecutar el siguiente comando en la consola de comandos en línea:

```
1 C:\Users\luispc>pip3 install virtualenvwrapper-winip3
```

Luego, se puede crear un nuevo entorno virtual con el comando mkvirtualenv.

CREACIÓN DE UN ENTORNO VIRTUAL

Posterior a instalar **virtualenvwrapper-win**, crear y trabajar con entornos virtuales es muy similar en todas las plataformas.

Procedemos a crear un nuevo entorno virtual con el comando mkvirtualenv. A medida que se ejecuta este comando, verás que se va adecuando en marcha el entorno. Al completarse, el nuevo entorno virtual estará



0

CREACIÓN DE UN PROYECTO DJANGO

activo, y se puede comprobar porque el comienzo del prompt será el nombre del entorno entre paréntesis (como se muestra abajo).

Una vez que estás dentro del entorno virtual, ya puedes instalar Django e iniciar el desarrollo.

USO DE UN ENTORNO VIRTUAL

Hay sólo otros pocos comandos útiles que deberías conocer. Existen más en la documentación, pero éstos son los que usarás de forma habitual:

- deactivate: salir del entorno virtual Python actual.
- workon: listar los entornos virtuales disponibles.
- workon nombre_del_entorno: activar el entorno virtual Python especificado.
- rmvirtualenv nombre_del_entorno: borrar el entorno especificado.

Procedemos a crear un proyecto en Django, con el nombre de: site_web_django.

```
1 (project_django) C:\Users\luispc>django-admin startproject site_web_django
```

Ejecutamos el servidor:

```
1 (project_django) C:\Users\luispc>cd site_web_django
2 (project_django) C:\Users\luispc>python manage.py runserver
```