Curso de Python: PIP y Entornos Virtuales

¡No te rindas!

Necesitas una **calificación mínima de 9.0** para aprobar.
Vuelve a intentarlo en 05 horas, 56 minutos, 12 segundos

7.5
Calificación

9 / 12
Aciertos

1. ¿Qué herramienta nos permite trabajar con entornos virtuales en Python 3?

pip REPASAR CLASE

2. ¿Qué herramienta nos permite instalar paquetes de Python como dependencias en nuestros proyectos?

pip ✓

3. ¿Con qué comando activamos entornos virtuales en Python 3?

source [ruta del entorno virtual]/bin/activate ✓

4. ¿Qué herraminta nos permite aislar y encapsular nuestros proyectos y los paquetes de terceros que este utilice, aunque la versión de Python y el sistema operativo sigan siendo los mismos para todos los proyectos?

venv ✓

---

5. ¿Qué herramienta nos permite aislar y encapsular TODO: archivos del proyecto, paquetes y sus versions, Python e incluso el sistema operativo?

Docker ✓

---

6. Tienes un archivo de texto llamado requirements.txt con todos los paquetes que necesitas con cierta versión en particular. ¿Cómo los instalas todos leyendo el archivo de texto con un solo comando?

pip3 install -r requirements.txt ✓

---

7. ¿Con qué comando guardamos el nombre y versión de todos los paquetes de terceros en nuestro proyecto dentro de un archivo de texto?

pip3 freeze > requirements.txt ✓

---

8. ¿Con qué comando instalamos el paquete requests en su versión 2.27.1?

pip install requests -v 2.27.1 REPASAR CLASE

---

9. ¿Con qué comando instalamos el paquete FastAPI en su última versión estable (sin importar la fecha en que se ejecute ni las viejas o nuevas versiones que se

desarrollen de FastAPI).

pip install fastapi -v last

REPASAR CLASE

---

10. Git y GitHub son herramientas indispensables para trabajar en equipo cuando usamos Python.

Verdadero ✓

---

11. En tu proyecto A necesitas matplotlib en su versión 3.5, pero tu proyecto B necesita el mismo paquete en su versión 3.6. ¿Cuál es la mejor forma de trabajar para no generar conflictos entre ambos paquetes?

Aislando cada proyecto en su propio ambiente virtual para instalar la versión correcta del paquete en cada uno sin afectar al otro. ✓

---

12. Estás desarrollando un proyecto en Python que utiliza diferentes paquetes de terceros en versiones muy especificas. ¿Cuál es la mejor forma de trabajar con el resto de mi equipo para que siempre instalen esos paquetes en esas versiones cuando clonen el proyecto?

Creando un requirements.txt con todos los paquetes y sus versiones e indicando en el README que instalen las dependencias leyendo ese archivo con pip. ✓

REGRESAR