Virtual Environment

Hvad er et Virtual Environment?
Et virtual environment (virtuelt miljø) er en isoleret mappe, der indeholder en separat Python-installation for en bestemt version af Python sammen med en samling af de nødvendige pakker. Dette gør det muligt at arbejde på forskellige projekter, hvor hvert projekt har sin egen unikke opsætning af Python og afhængigheder uden at påvirke systemets globale Python-installation. Dette er især nyttigt, når man arbejder på flere projekter, der kræver forskellige versioner af de samme biblioteker.

Sådan fungerer et Virtual Environment
Når et virtual environment er aktivt, sørger det for, at alle pakker og afhængigheder, der bruges, er specifikke for det aktive miljø og ikke konflikter med andre projekter. Dette opnås ved at ændre stien (`sys.path`), så Python kun bruger det aktuelle miljøs biblioteker og pakker. Dette sikrer, at alle Python-programmer, der startes i det, bruger den Python-version og de pakker, der er installeret i det aktive miljø, hvilket giver fuld kontrol over, hvilke pakker der bruges til hvert projekt.

Aktivering og Brug
Når miljøet er aktiveret, vil alle Python-programmer, der startes i det, bruge den specifikke version og samling af pakker i det pågældende virtual environment. Python-programmer kan også køres i et specifikt virtual environment ved at aktivere det forud, hvilket gør det muligt at køre flere programmer samtidig med forskellige samlinger af afhængigheder. Dette er særligt nyttigt i projekter, hvor forskellige versioner af de samme biblioteker er nødvendige. På den måde kan udvikleren sikre, at afhængighederne i ét projekt ikke har nogen indflydelse på afhængighederne i et andet projekt.
For en konkret guide til opsætning og brug af virtuelle environments, så se dokumentet *Package Management Python*.

Hvorfor skal du bruge Virtual Environments?

Isolation
Virtual environments sikrer, at hvert projekt har præcis de afhængigheder, det har brug for, uden at dele dem med andre projekter. Dette eliminerer risikoen for konflikter mellem afhængigheder, såsom når to projekter kræver forskellige versioner af det samme bibliotek. Denne isolation beskytter ikke kun projekternes stabilitet men sørger også for, at den globale Python-installation ikke bliver "forurenet" med uønskede pakker, hvilket kan føre til kompatibilitetsproblemer på systemniveau.

Reproducerbarhed
Virtual environments gør det nemt at eksportere projektets afhængigheder ved hjælp af en `requirements.txt` fil, som indeholder en liste over alle pakker og deres versioner, der er installeret i miljøet. Dette betyder, at andre udviklere nemt kan genskabe det samme miljø ved blot at installere pakkerne fra denne fil. Denne reproducerbarhed er essentiel for teamsamarbejde og vedligeholdelse, da det sikrer, at alle arbejder i det samme præcise miljø, hvilket mindsker risikoen for fejl relateret til forskelle i softwareversioner.

Fleksibilitet
Med virtual environments kan udviklere eksperimentere med forskellige versioner af pakker uden at påvirke det overordnede udviklingsmiljø. Hvis man f.eks. vil teste en ny version af en bestemt afhængighed, kan dette gøres uden at ændre de versioner, som andre projekter afhænger af. Dette giver fleksibilitet til at prøve nye løsninger og foretage opdateringer uden risikoen for at ødelægge andre projekters funktionalitet.

Simpelt Setup for Nye Udviklere
Virtual environments gør det meget nemmere for nye udviklere at komme i gang med et projekt. De kan hurtigt oprette et lokalt miljø med de korrekte afhængigheder ved hjælp af `requirements.txt`. Dette eliminerer tidsspild og frustration, som kan opstå ved at forsøge at installere alle nødvendige pakker manuelt og sikre, at de korrekte versioner er tilgængelige. Det gør onboarding-processen mere gnidningsfri, hvilket sparer tid og reducerer fejl, der skyldes forkerte opsætninger.

Ressourcer

Her er en samling af forskellige ressourcer som kan bruges i forbindelse med læring om- og brug af virtual environments.

- Kondenseret SPAC guide om brug af Virtual Environments: Package Management Python
- Officiel Python Dokumentation om **venv:** https://docs.python.org/3/library/venv.html
- Officiel **venv** tutorial: https://docs.python.org/3/tutorial/venv.html
- Real Python Guide til Virtual Environments: https://realpython.com/python-virtual-environments-a-primer/
- Python Packaging Authority's Vejledning. En guide til at bruge pip og virtualenv, en alternativ pakke til venv: https://packaging.python.org/guides/installing-using-pip-and-virtual-environments/
- Install packages in a virtual environment using pip and venv: https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments