Package Management Python

Hvis du allerede kender til Python virtuel environment (venv) eller pipfile, kan du se bort fra dette

Pip

Pip er Pythons egen packet manager som downloader packages lokalt på brugerens computer. Et problem kan dog opstå hvis man arbejder i et større projekt hvor der er måske 10 dependencies. Fordi der er så mange forskellige biblioteker i pip, og mange gør lidt det samme, kan der hurtigt opstå et konflict i packages. Hertil har Python en ret kendt løsning med deres venv.

Virtual environment (venv)

Dette område handler kun præcist om hvordan man bruger og opsætter et virtuelt environment via en terminal. Hvis du vil lære om hvad præcist et virtuel environment er så kan du kigge i dokumentet Virtual Environment.

Måden man laver et venv i Python er heldigvis ret simpel ved at man i en terminal skriver:

```
python -m venv .venv
```

Kommandoen gør følgende:

1. Kalder python.exe

2. Kalder standard biblioteket venv

3. Laver et nyt venv i mappen der her er kaldet .venv (man kalder generelt altid mappen .venv men i princippet kan den hede alt)

Ende i .venv mappen vil der komme til at ligge en lokal instans af pip og dermed de biblioteker man downloader, samt en lokal instans af Python.

For at komme ind i venv'et gør man følgende i Windows:

```
.\.venv\Scripts\activate
```

Eller i Linux/macOS:

```
. .venv/bin/activate
```

Dette vil typisk tilføje et (.venv) i den terminal for at tydeliggøre at du er i et venv. Nu kan du bruge pip install som du plejer.

En anden vigtig ting er at lave en såkaldt requirements.txt fil. Her vil der stå hvilke dependencies der er påkrævet for at køre programmet. De kan installeres ved at køre i terminalen:

```
pip install -r requirements.txt
```

ellers kan den laves/opdateres ved at skrive:

```
pip freeze > requirements.txt
```

Det er vigtigt at man har installeret de dependencies der allerede ligger i requirements inden man opdaterer den, da den i virkeligheden overskriver filen