Humanistens Digitale Værktøjskasse

Indledning

Denne tekst er et overblik over hvad der gennemgås i workshoppen "Humanistens Digitale Værktøjskasse". Dette dokument forsøger at give et overblik over hvordan de forskellige elementer af workshoppen kan bruges også efter workshoppen er slut. Den er altså tænkt, som et referencepunkt, man kan vende tilbage til, når man er i tvivl om hvordan man anvender materialet fra workshoppen.

Hvad skal man bruge til workshoppen?

Til workshoppen skal vi hente to programmer: Miniconda og Visual Studio Code.
Miniconda er det program, som vi skal bruge til at køre vores kode i. Det består af følgende tre ting: kodesproget python, en pakke-manager og en miljø-manager. Dette kan lyde forvirrende. Kort fortalt, er det den måde, vi installere kodesproget, som vi bruger under workshoppen på vores computere.
Visual Studio Code er det program, som vi bruger til at ændre i vores kode. Det skaber et mere overskueligt overblik, så vi kan følge med i hvad der sker. Visual Studio Code giver vores kode farver og gør det mere overskueligt, i stedet for at man kigger på det i computerens indbyggede textEditor.

Hvordan åbner jeg miniconda?

Miniconda (fremover conda) åbnes forskelligt alt efter om du bruger Windows eller Mac.

Windows

På Windows skal vi åbne programmet Anaconda powershell promt (miniconda), som ADMINISTRATOR. Dette Command Line Interface (CLI) skal vi arbejde i under workshoppen. Det er en anden måde at arbejde med sin computer på, som kan kræve lidt tilvænning og oplæring

Mac

På Mac, skal man åbne programmet ”terminal” efter installationen af miniconda. Dette kan gøres ved at bruge genvejen cmd + mellemrum og så søge "terminal". Terminalen er Mac-systemets Command Line Interface (CLI) som vi skal arbejde i under workshoppen. Det er en anden måde at arbejde med sin computer på, som kan kræve lidt tilvænning og oplæring

Opsætning af conda

Første gang man bruger conda, skal man lave det miljø, som man vil arbejde i. Det er her man sikre sig, at forskellige python projekter ikke "ødelægger" hinanden.

For at lave et nyt miljø i conda skriver man følgende command;

```
conda create --name "toolbox" python=3.8
```

Når dette miljø er lavet, skal man aktivere det. Det gør man ved at skrive:

```
conda activate "toolbox"
```

Dette er det grundlæggende conda, som vi bruger i løbet af workshoppen. Hvis man efter at have lavet miljøet "Toolbox" lukker vinduet ned, skal man aktivere sit miljø igen.

Nedenfor er en liste af brugbare conda commands:

conda create - -name "name" python=3.8	Laver et nyt miljø i conda, som kører python 3.8, der hedder name
conda activate "name"	Aktiverer conda miljøet name
conda deactivate	deaktiverer det aktive conda miljø
conda install	installerer pakker til conda
conda uninstall	afinstallerer pakker i conda
conda update	Opdaterer conda
conda update "package"	Opdaterer pakken "package"

Sådan navigerer man i Shell og Terminal

Når man bruger programmerne Shell (Windows) & Terminal (Mac) interagerer man med computeren på en helt anden måde. For det første, kan man tænke på sig selv som en sej hacker, eller som en computernørd fra start 80'erne, før computere fik det udseende, som vi interagerer med den dag idag. Grunden til at vi skal denne vej ind, er fordi vi denne vej kan få lov til mange andre ting med computeren. Men for at det kan lade sig gøre, skal vi vide hvordan vi bevæger os rundt i systemet, når vi ikke har en mus til at klikke på mapperne. Nedenfor har jeg indsat en tabel, der indeholder nogle af de mest brugbare commands til at finde rundt i Shell og Terminal

Command:	**Forklaring**:
cd	Command der skifter mappe, den skal efterfølges af hvor man gerne vil hen i sin computer.
cd ./mappe	Skifter til mappen "mappe" under den mappe, som vi befinder os i nu.
cd ..	Skifter til mappen ovenover den vi er i nu.
cd ~/	Skifter til computerens hjemmeplacering, som kan være et godt udgangspunkt.
Cd c:/users/name/billeder	Skifter til mappen billeder under name, under users på c-drevet.
dir	viser os indholdet af den nuværende mappe
ls	viser os indholdet af den nuværende mappe

Det første script i workshoppen - rename.py

Dette script kan bruges til at omdøbe en masse filer, der har den samme endelse, på en gang. I workshoppen bruger vi scriptet på billeder fra Vikingemuseet i Århus, det kunne også bruges på fx scannede dokumenter, kilder eller noget helt tredje.

I dette script er der to ting, som man skal ændre på, for at få det til at virke. Først åbner man filen rename.py i Visual Studio Code. De to steder, som man skal ændre er i linje 8 og linje 11.

I linje 4, skal man specificere hvor filerne er placeret. Det kunne fx være, at jeg havde mine billeder til at ligge i mappen 1Billeder, i mappen "Humanistens_digitale_toolbox" på mit skrivebord. Så ville linje 4 se sådan her ud:

path="C:/Users/user/Desktop/Humanistens_digitale_toolbox/1Billeder/"

OBS: Når man skriver denne path, er det vigtigt at huske at slutte den af med en "/" og for windows-brugere, skal man vende alle "\", så de bliver "/".

I linje 12, skal man skrive det navn, som filerne skal have + den endelse, som alle filerne ALLEREDE har. Navnet skrives istedet for name_of_files og filendelsen istedet for .endning. Under workshoppen ville vi gerne have vores .jpg-billeder til at hedde: vikingemuseet:

os.rename(path + file, path + "vikingemuseet_{}".format(i)+".jpg")

OBS: Dette script ændrer alle endelser i mappen til den man har specificeret i linje 12. Det kan IKKE bruges til at ændre filernes endelser. Har man filer i mappen, der har en anden endelse ødelægges disse filer muligvis i processen. Vi foreslår at man laver en sikkerhedskopi af mappen, før man anvender scriptet

For at køre scriptet, skal man i powershell (windows) / Terminal (mac) først finde hen til det. Dette gøres med commanden cd efterfulgt af pathen til filen. Det kan fx se sådan her ud:

cd C:\\Users\\user\\Desktop\\Humanistens_digitale_toolbox\\Scripts

OBS: På windows bruger man \. På Mac skal man bruge /.

Når man tror man har fundet den rigtige placering, kan man tjekke ved at bruge enten commanden "dir" eller "ls". Hvis scriptets navn kommer frem på skærmen, er man klar til at køre filen, dette gøres ved at skrive:

python rename.py

Hvis man ingen fejlkode får, er filerne omdøbt. Hvis man modtager en fejlkode, så er det højst sandsynligt fordi man enten har glemt et " eller lavet et mellemrum et sted, så kig koden igennem igen i Visual Studio Code.

Det andet script i workshoppen - Selectivecopy.py

Dette script kan kopiere mange filer på en gang. I workshoppen bruges det til at kopiere forskellige rapporter fra flere mapper til en samlet mappe. Scriptet virker på alle filer der har samme endelse. Jeg bruger det fx mod slutningen af et semester, til at samle alle tekster fra et kursus til en samlet mappe.

I dette script skal man lave ændringer i linje 9,10, 24 & 25. I linje 9 skal man indsætte pathen til der hvor filerne befinder sig og i linje 10 skal man indsætte pathen til der filerne skal kopieres til.

Linje 9 kommer til at se nogenlunde sådan her ud:

source = "C:/Users/user/Desktop/Humanistens_digitale_toolbox/2Rapporter/"

OBS: Husk / tilsidst og for windowsbrugere at vende \ til /.

Linje 10 kommer til at se sådan her ud:

source = "C:/Users/user/Desktop/Humanistens_digitale_toolbox/2Rapporter/tekster/"

Hvis den nye placering er en ny mappe, på samme placering som filerne befinder sig, er det vigtigt at ændre linjerne 24 og 25. Her skal man ændre navnet på mappen til det navn, som man kalder den nye mappe

```
if 'tekster' in subfolders:
    subfolders.remove('tekster')
```

Dette gøres for at scriptet ikke skal kopiere alle filerne to gange.

Det tredje script i workshoppen – pdfannots.py

Det tredje script, som vi kommer til at bruge i workshoppen hedder pdfannots.py. Dette script skal vi bruge på en anden måde end hvad vi har gjort med de første to. Scriptet er udviklet af Andrew Baumann, som bruger det, når han reviderer akademiske artikler og konferencebidrag, Det kan dog også hjælpe os andre der læser PDF-filer. Scriptets kernefunktion er, at trække kommentarer og overstregninger ud af PDF-filer. Lad os kigge på det:

Hvis man åbner dette script i visual studio code opdager man, at det er MEGET længere end de andre, som vi har arbejdet med. Heldigvis skal vi ikke redigere i scriptet på samme måde som tidligere. Hele interaktionen med scriptet foregår i vores CLI.

Det første vi skal gøre i command linen er at installere et modul/en pakke i vores værktøjskasse. Vi skal bruge et modul der hedder pdfminer.six. Dette installeres ved at skrive pip install pdfminer.six og trykke enter. Et modul er en samling af kode, som andre allerede har skrevet, så vi er fri for at skabe det hele fra bunden af igen. Vi står altså på andre kodebyggeres skuldre.

Når vi skal bruge scriptet, skal vi gøre følgende i vores CLI:

1. Først naviger hen til scriptet i terminalen
2. Dernæst aktiver scriptet ved at skrive: `python pdfannots.py path_til_pdf_der_skal_trækkes_ud_fra`

Dette vil give alle annotationer direkte i det vindue man sidder med og man kan så kopiere teksten videre til Word, OneNote, Notion, EverNote eller hvilket som helst program, som man foretrækker at samle sine notater i. En anden mulighed er, at gemme den udtrukne tekst i en tekstfil, Dette kan gøres på følgende måde:

1. Naviger hen til scriptet i terminalen
2. Aktiver scriptet ved at skrive: `python pdfannots.py -o [Navnet_på_den_txt_man_vil_have_resultat_i.txt] path_til_pdf_der_skal_trækkes_ud_fra`

En sidste mulighed, er at navigere hen til scriptet og skrive `pdfannots.py -h`. Dette vil vise scriptets indbyggede menu, hvor man kan se hvilke muligheder scriptet indeholder