En kort introduksjon til Jupyter Notebook

I dette dokumentet gir vi en kort innføring til *Jupyter notebooks*, som er et verktøy man kan bruke til å skrive og kjøre kode - det er altså et alternativ til å bruke Spyder eller andre teksteditorer. Vi vil her gi en overordnet forklaring på hva Jupyter er, hvordan det brukes og hvorfor det kanskje kan være et nyttig og interessant verktøy.

Hva er Jupyter notebook?

Når vi programmerer må vi skrive programkode, dette gjør vi i en teksteditor. Når vi har skrevet koden har vi et program vi kan kjøre for å få ting til å faktisk skje. Spyder er et eksempel på en teksteditor av denne typen, og lar oss kjøre kode i samme program som vi skriver koden. Dette er fint, fordi man kan skrive kode, prøve den, så endre på den. Dette gjør det lettere å finne å skrive kode, samt å utforske den etterpå.

En ulempe ved å bruke en tradisjonell teksteditor som Spyder derimot, er at det er vanskelig å kombinere koden og resultatene våres med annen tekst, figurer og matematikk.

Det er her Jupyter kommer inn i bilder. Jupyter er ikke en tradisjonell teksteditor, istedet er det en *notebook*. I disse notebooksene kan man fritt blande kode med annet innhold som tekst, figurer, matematikk, tabeller, og lignende. Samtidig kan kjøres koden og man får resultatet rett inn i samme dokument som koden vår. På denne måten kan notebooks gi et mer helhetlig produkt enn å bruke en editor som Spyder.

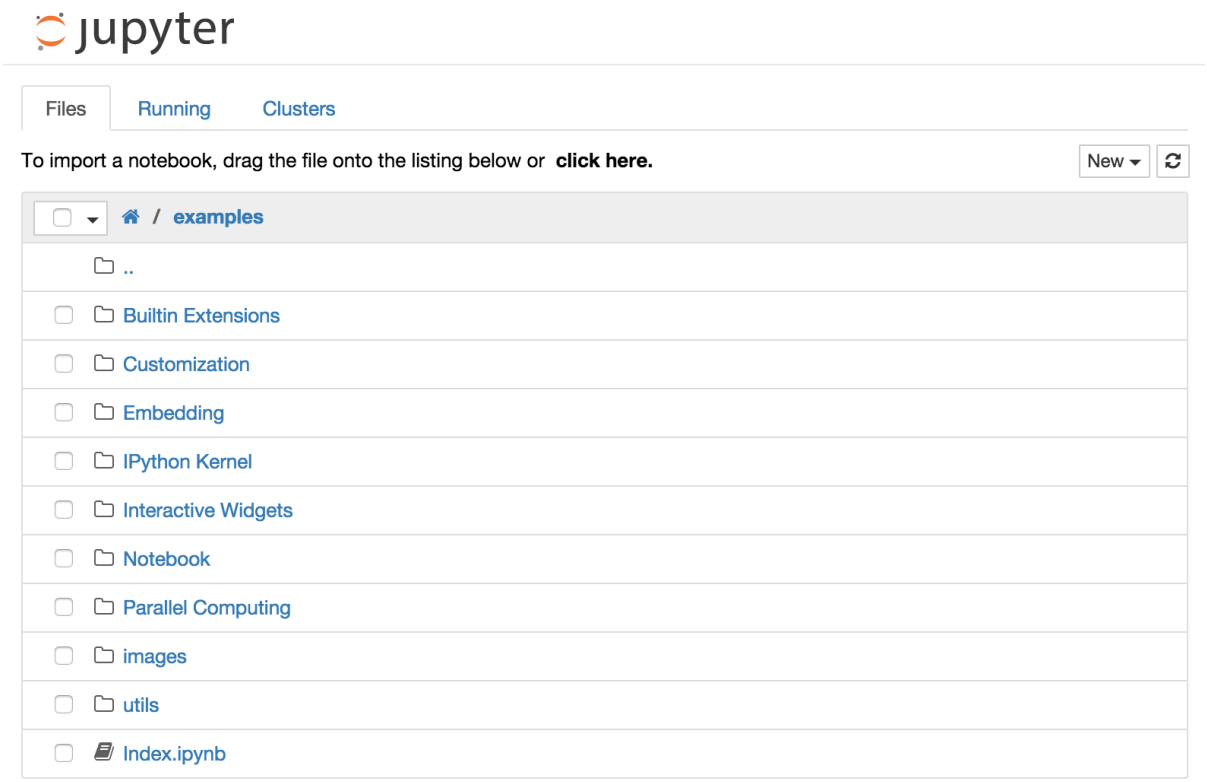
Hvordan installere og åpne Jupyter?

Jupyter er programvare som må installeres for å brukes. Den enkleste måten å installere Jupyter på er å installere programpakken *Anaconda*. Ved å laste ned Anaconda får du en Python-installasjon, Spyder, og tileggspakker som pylab installert alt på én gang.

Om du har installert Anaconda på maskinen din skal du kunne starte programmet som heter 'Jupyter Notebook'. Om du ikke finner dette programmet, prøv isteden å starte 'Anaconda Navigator' og bruk denne til å starte Jupyter. Alternativt, om du vet hvordan du åpner en terminal kan du gjøre det og skrive inn "Jupyter Notebook".

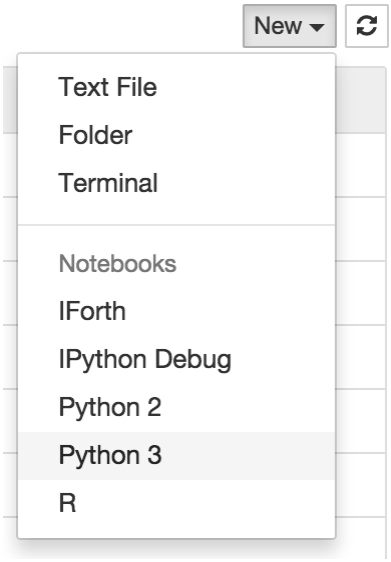
Når du starter opp Jupyter-programmet, vil datamaskinen begynne å kjøre en slags server i bakgrunnen. Samtidig åpner den din standard nettleser for å interagere med denne serveren. Når du jobber i Jupyter jobber du altså i din vanlige browser, for eksempel Google Chrome, Safari, eller Edge. Selv om du jobber i nettleseren bruker du ikke internet, og Jupyter vil fungere fint selv om du er frakoblet internet.

Når du først åpner Jupyter vil du være på det som heter *Dashbordet*. Dette er en navigasjonsmeny du kan bruke for å finne frem til den notebooken du ønsker å åpne, eller som du kan navigere og lagre nye filer.



Figur: Når du åpner Jupyter vil du først møte dashbordet, som lar deg navigere frem til de filene du ønsker, eller lage nye filer.

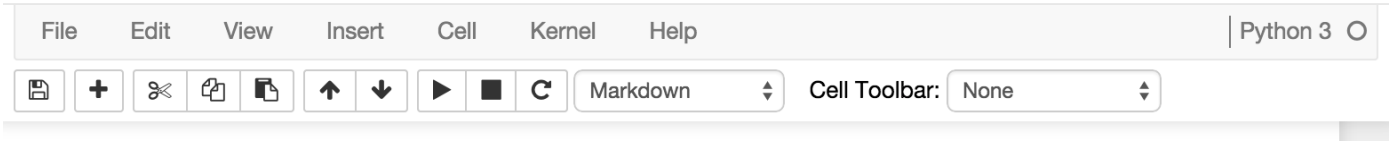
Siden dette er første gang du bruker Jupyter kan vi opprette en ny, og tom, notebook. Dette kan du gjøre ved å klikke på knappen **New** øverst til høyre. Jupyter kan brukes i mange forskjellige programmeringsspråk, så velg Python3.



Hvordan bruker vi Jupyter?

Om du har klart å lage en ny fil vil skjermen din nå stort sett være tom. Dette er fordi vi foreløpig er i et tomt dokument.

På toppen av skjermen har vi en verktøylinje vi kan bruke til å navigere og endre på dokumentet vårt:



I tilegg til verktøylinja på toppen vil dokumentet foreløpig kun bestå av én tom celle, som ser slik ut:

```
In [ ]:
```

I Jupyter består et dokument av en rekke slike *celler*, som rett og slett er blokker med innhold. Vi deler alle slike celler i to: tekstceller og kodeceller. I tekstceller kan vi ha alle typer innhold som ikke har noe med koden å gjøre, f.eks brødtekst, matematikk, figurer og tabeller. I kodecellene kan vi skrive Pythonkode, og vi kan også kjøre kodecellene og se resultatet i notebooken.

Når du jobber i en notebook vil du alltid være i én av to moduser: Enten jobber du aktivt i én bestemt celle, for eksempel skriver du kode. Dette kalles "edit-mode", eller redigeringsmode. Når du er i redigeringsmode vil cellen du jobber i være markert med en grønn boks, og du har en blinkende tekstmarkør som indikerer at du kan skrive og gjøre endringer.

```
In [1]: a = 10
```

Om vi ikke aktivt jobber på en celle kan vi istedet navigere i dokumentet, lage nye celler, kjøre celler, kopiere celler og så videre. Eller kanskje vi bare leser dokumentet. Dette kalles "Command mode", eller bare navigeringsmodus. I denne modusen er den nåværende valgte cella markert med et blått-vindu:

```
In [1]: a = 10
```

Om du er i navigeringsmodus og ønsker å redigere en celle kan du dobbeltklikke på den, eller du kan navigere til den med piltastene og trykke "Enter". Når du er ferdig å redigere en celle kan du rett og slett trykke utenfor cella, eller trykke "Escape"-tasten.

Oppgave 1) Hello, World!

Det er på tide at du prøver å skrive og kjøre en kodecelle. Gjør følgende steg

- Om du ikke allerede har gjort det. Åpne Jupyter Notebook og lag en ny notebook fil
- Velg den eneste cella som er i det nye dokumentet
- Skriv inn Pythonkode som skriver ut beskjeden *"Hello, World!"*
- Kjør cella ved å trykke på "Run"-knappen i verktøylinja på toppen.

Om du følger disse stegene og alt fungerer som det skal vil du få et resultat som ser slik ut:

In [ ]:

```
print("Hello, World!")
```

Her ser vi at kodecella har blitt kjørt, fordi det står "In [1]" til venstre for cella. Dette betyr at det er den første cella som har blitt kjørt i notebooken så langt. Under cella kommer resultatet av kjøringen. I Notebooks vil outputten fra en celle alltid komme rett under selve cella.

- Prøv nå å gå tilbake å endre koden og kjør på nytt. Hva skjer?

Merk at du også kan kjøre celler ved å trykke `Ctrl+Enter` eller `Shift+Enter`.

Lage nye celler og manipulere celler

En kodecelle kan godt ha flere linjer med kode, vi kan for eksempel lage følgende celle:

In [ ]:

```
s0 = 1.5
v0 = 12
a = -9.81
t = 1.2

s = s0 + v0*t + 0.5*a*t**2

print(f"s({t:.1f}) = {s:.1f}")
```

Det er ingen begresning på hvor stor en gitt celle kan være. Derimot er det ofte hensiktsmessig å dele koden vår over flere celler. Om man løser oppgaver kan man for eksempel ha en kodecelle per oppgave.

Du kan bruke verktøylinja på toppen til å lage nye celler, slette celler du ikke vil ha, kopiere celler eller bytte på rekkefølgen.

Om kjøring av celler

Et element av notebooks som kan være litt forvirrende når man kommer fra en tradisjonell editor som Spyder er at en gitt notebook er én sammenhengende interaktiv sesjon. Det betyr at variabler som defineres i en celle vil huskes til neste celle.

Om man deler en notebook som består av mange celler kan det være nødvendig å kjøre celler fra toppen av dokumentet og nedover. En nyttig funksjon her er å gå på `Kernel` i verktøylinja og velge `Restart and Run All`. Dette betyr at alle celler kjøres fra start av dokumentet og nedover.

Lage og redigere tekstceller

I tilegg til kodecellene kan vi lage celler som har annet innhold. Disse kalles for *Markdown*-celler i Jupyter, men vi kan kalle dem for tekstceller. Markdown er en type tekstformattering som brukes mye på nett, for eksempel i kommentarfelt, i blogger og på forum.

For å lage en tekstcelle lager du en ny celle som vanlig, men deretter endrer du typen til Markdown ved å velge cellen og klikke på type-valget på verktøylinja. Her står det "Code" for en kodecelle, og du velger istedet "Markdown".

Når du har endret cellen til markdown kan du redigere cella akkurat som om det var en kodecelle. Men nå kan vi ikke kjøre cella på samme måte, derimot vil cella vises frem som pen brødtekst om du "kjører" den.

Tekstformattering med Markdown

I tilegg til å bare skrive ren tekst kan vi legge på spesielle symboler for å få enkel tekstformattering. Det er her *Markdown* kommer inn i bildet. Her kan du for eksempel bruke `*` for å få kursiv, eller `**` for å få fet skrivt. For eksempel blir `*Jupyter*` til *Jupyter*.

Om vi ønsker å skrive ut variabelnavn eller kodelinjer i teksten vår som skal formateres som kode kan vi bruke apostrofer:

- `print("Hello, World!")`

blir til

- `print("Hello, World!")`

Dette kan være fint å bruke når man skal referere til konkrete funksjoner eller variabler. For eksempel: "I Python så vil `print`, skrive ut tekst til skjermen."

Markdown er en annen tankegang å måte å jobbe på en for eksempel Word, og det kan være litt uvant i starten, men man blir fort vant til det og man kan skrive tekster raskt og effektivt. Merk også at det ikke er viktig å kunne markdown for å lage en notebook som kjører som den skal, det er mest for kosmetiske grunner. Vi går nå igjennom en rekke funksjonalitet vi kan få til med Markdown, og ikke alt er like viktig, det er mest ment som et referanseverk.

Overskrifter

Man kan lage overskrifter ved å starte linjen med én eller flere `#`. Jo fler man bruker jo mindre blir overskriften, og man kan bruke opp til 6. For eksempel:

Overskrift nivå 2

```
## Overskrift nivå 2
```

Overskrift nivå 4

```
#### Overskrift nivå 4
```

Matematikk i Jupyter

Matematikk og programmering er knyttet tett sammen, og vi ønsker ofte å inkludere matematikk i notebookene våres - dette finnes det god støtte for. Jupyter bruker noe som heter *MathJax*, som er laget for å vise matematikk i browsere. For å skrive matematikk bruker vi samme kommandoer som i *LaTeX*, om du kjenner til dette.

Kort fortalt bruker man `$` for å indikere at man skriver matematikk. Om vi bruker enkle dollartegn sier vi det er *inline* matematikk, det holder seg altså på samme linje som resten av teksten. For eksempel blir `$a = \pi r^2$` om til $a = \pi r^2$. Bruker vi doble dollartegn, så vil det istedet blir formatert på en egen linje `$$ a = \pi r^2$$` blir til:

$$a = \pi r^2$$

Som du ser fra eksemplene kan vi bruke `\pi` for å få greske bokstaver og `^` for opphøyd i, vi kan bruke `_` index, `H$_2$0` blir til H_2O. Merk at alle symboler i matte uttrykk automatisk blir kursiv, dette er fordi variabler i matematikk skal være i kursiv. Om vi ønsker å ha tekst, kan vi bruke `\text{}` rundt teksten vår

$$\text{areal} = \pi r^2.$$

Vi kan bruke `\frac{}{}` for å lage brøker, der den første parantesen blir nevner og den andre blir telleren

$$\text{areal av kule} = \frac{4\pi r^3}{3}.$$

Vi kan bruke `\sqrt{}` for kvadratrot:

$$a^2 + b^2 = c^2 \quad \Rightarrow \quad c = \sqrt{a^2 + b^2}.$$

Her kan vi også nevne at matematikk på Wikipedia vises frem med samme teknologi som ligger bak Jupyter. Om du markerer en matematisk formel på Wikipedia så can du rett og slett Copy+Paste denne rett inn i en Jupyter tekst-celle (husk å legge til dollartegn), så har du formelen enkelt som bare det.

For eksempel kan vi gå inn på Wikipedia sin artikkel om Binomialdistribusjonen (https://en.wikipedia.org/wiki/Binomial_distribution) og kopiere over binomialkoeffisienten:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Nettsider

Ettersom at Jupyter bruker nettleseren til å redigere på dokumentet vårt føles det veldig naturlig å bruke hyperlenker for å hoppe til andre nettsider. Dette kan vi gjøre som følger:

```
[lenketekst](nettaddresse)
```

Der nettaddressen ikke vises til brukeren, men man kan se den om man holder over linken uten å klikke på den.

Eksempel:

```
 Som du kan lese i for eksempel [Aftenposten]
(https:www.aftenposten.no)
```

Blir til

> Som du kan lese i for eksempel Aftenposten (https://www.aftenposten.no)

Her er det for eksempel ofte nyttig å lenke til nettsider der man har innhentet informasjon, eller hvor man kan lese mer om et tema.

Bilder

Vi kan legge inn bilder i tekstceller. For å gjøre dette bruker vi HTML istedetfor Markdown (Jupyter skjønner HTML godt). Syntaksen for å vise et bilde i HTML er som følger:

```
<img src="bildeaddresse">
```

Der bildeaddressen enten kan være en nettadresse, eller en filplassering på egen maskin. Om du i brosweren din høyreklikker et bilde på internet vil du kunne velge `Kopier Bildenettaddresse` eller noe lignende. Vi kan for eksempel gå på Wikipedia og finne et bilde av en rød panda (https://en.wikipedia.org/wiki/Red_panda#/media/File:RedPandaFullBody.JPG)

Om vi kopierer bildeaddressen vil bildet gjengis i original størrelse, som kan bli veldig stort eller veldig lite:

Men vi kan endre på størrelsen ved å gi bredden eller høyden som et ekstra argument (bildet skaleres likt i begge dimensjoner

```
<img src="..." width=400>
```

gir

Om bildet ligger på egen maskin, og ikke på nett, skriver vi filnavnet i `src`, og vi må ha med filendelsen. Om bildet ligger i samme mappe som notebookfilen kan vi bare skrive mappenavnet, ellers må vi ha med mappestrukturen. Om den for eksempel ligger i en mappe som heter "bilder" kan vi skrive

```
<img src="bilder/mitt_bilde.png">
```

Merk at når vi bruker HTML til vise bilder på denne måten så vil vi alltid refere til et bilde utenfor notebooken. Om de ligger på nett er dette greit, for alle med nettilgang vil se dem når de åpner notebooken. Med egne bildefiler derimot, må disse sendes med ekstra.

Man kan også legge til bilder i en notebook på en måte der de følger med notebooken ved å dra bildet og sleppe det i en markdown celle - men dette er ikke helt idéelt, da det tar mye mer plass og gjør at notebooken bruker mer tid på å laste inn. Samtidig får vi mindre kontroll over bildet, da vi for eksempel ikke kan velge størrelsen. Om man lager notebooks man ønsker å dele med andre kan det være en idé å laste opp bilder til nett først, og så linke til dem. Her kan man for eksempel bruke imgur (https://imgur.com/), men pass isåfall på at du har rett til å bruke bildene du laster opp, og at de ikke inneholder noe sensitivt.

Lister og Tabeller

Punktlister

Punktlister lages ved å starte hver linje med *

```
* Norge
* Sverige
* Danmark
```

Blir til

- Norge
- Sverige
- Danmark

Eller vi kan bruke tall

```
1. Norge
2. Sverige
3. Danmark
```

Blir til

1. Norge
2. Sverige
3. Danmark

Vi kan også lage enkle tabeller

```
| Land        | Innbyggere    | Areal            |
| ----------- |-------------  | -----------------|
| Norge       | 5.2 millioner | 385 tusen km$^2$ |
| Sverige     | 9.8 millioner | 447 tusen km$^2$ |
| Danmark     | 5.7 millioner | 43 tusen km$^2$  |
```

Blir til

Land	Innbyggere	Areal
Norge	5.2 millioner	385 tusen km^2
Sverige	9.8 millioner	447 tusen km^2
Danmark	5.7 millioner	43 tusen km^2

Youtube-videoer

Om du ønsker å legge inn en youtube-video i notebooken din kan du gjøre dette som en enkel nettlenke:

- Link til Youtube-video: "Code Stars" (https://www.youtube.com/watch?v=dU1xS07N-FA)

Eller du kan "embedde" videoen rett inn i notebooken. For å gjøre dette, gå på Youtube og klikk på "Share" knappen under videoen og velg embed. Der vil du få en en tekst-streng med HTML kode du kan kopiere inn i notebooken din. Da viser du koden din frem med en enkel Python kode som følger:

In [2]:

```
from IPython.display import HTML
HTML('<iframe width="560" height="315" src="https://www.youtube.com/embed/dU1xS07N-
```

Out[2]:

Hvorfor velge å bruke notebooks? Fordeler og Ulemper

I dette kurset ønsker vi å fokusere på bruk av programming som et verktøy i realfaglige problemstillinger. Dette betyr at det er viktig å se koden vi skriver og programmene vi lager i en større kontekst. Jupyter notebooks gjør det mulig å kombinere koden inn i en større tekst eller raport på en naturlig måte.

Notebooks er hovedsakelig en god måte å utvikle og dele lærerressurser og opplegg, ettersom at man kan gi ut eksempelkode med forklarende tekst. Elever trenger da ikke å skrive over kode eller bruke noe tid på å klippe og lime, de kan gå rett inn i eksempler og endre og utforske. Med tradisjonelle læremidler kan man vise eksempelkode, men det kan være vanskligere å forstå for leseren når man ikke får kjørt programmene eller endret noe.

Notebook kan også være et godt verktøy for å jobbe prosjektbasert. Om man for eksempel skal lage en kort rapport av et programmeringsprosjekt i Word må man finne en måte å få koden sin i Word, for eksempel ved å ta et skjermbilde. Resultatene etter å ha kjørt koden må også inn, også gjerne som bilder. Om man ønsker å endre på koden sin og kjøre på nytt må man nå først endre koden, så få denne endringen inn i Word. Om de som så senere skal lese rapporten ønsker å dobbeltsjekke noe ved koden må de først skrive av koden og så kjøre den. En løsning på dette er at man leverer rapporten og koden hver for seg, men dette gjør det vanskligere å forstå konteksten. Kort fortalt kan det bli litt av en prosess. Med en notebook derimot kan elevene jobbe seg frem til én fil der koden inngår som en del av dokumentet. Man kan ha en introduserende tekst som beskriver problemstillingen, så kan man introdusere litt kode, så kan man drøfte resultatene, ha litt mer kode, osv. Om man ønsker å endre koden er det bare å gjøre dette og kjøre, og resultatene i rapporten oppdateres automatisk. Til slutt får man én enkelt fil man kan dele med andre, og de som leser har mulighet til å lese, men også å kjøre koden.

Dersom man skal dele dokumentet sitt med noen som ikke har eller ønsker å installere Jupyter, eller man ønsker å skrive ut til papir, så kan Jupyter enkelt lage .pdf-filer som kan deles med alle. Den enkleste måten å gjøre dette er rett og slett å printe dokumentet med `Ctrl+P` og velge "Lagre som pdf"/"Save to pdf", her kan man selvfølgelig skrive ut til papir om man ønsker også.

Jupyter brukes over hele verden

Jupyter notebook er bare ett av mange verktøy som lages av Jupyterprosjektet. Selve prosjektet er open source, som betyr at de deler all programvaren og kildekoden åpent og gratis for alle. Verktøyene som utvikles av Jupyter, inkludert notebbok, brukes idag av mange store tekonologibedrifter som for eksempel Google, Microsoft, IBM og NASA. Veldig nylig ble Jupyter tildelt en gjev softwarepris (ACM Software System Award), som deles ut for programvare som ser en bred brukerbase og som fører til en varig forbedring av IKT over hele verden. Det er mange kjemper blant tidligere vinnere, som for eksempel WWW (world wide web), Java (et populært programmeringsspråk), UNIX (grunnlaget for Linux og Mac OS). Det er altså tydlig at Jupyter og Jupyter notebook er programvare som er av høy kvalitet og som ser ut til å bli mer og mer populært over tid.

Det er ikke bare innen IKT sektoren at Jupyter ser mye bruk, det er også meget populært i realfagene, både til forskning og undervisning. Ved universiteter over hele verden utvikles og deles kursmaterialer i Jupyter notebooks, og forskere bruker Jupyter til å analysere, vise frem og dele forskningen sin. For eksempel har nobelprisvinnerene i Fysikk fra 2017 delt datasettene fra oppdagelsen de gjorde av gravtiasjonsbølger. Dette gjorde de åpent for alle, og valgt og gjøre det i Jupyter notebooks.

Tastatursnarveier

Det finnes en lang rekke snarveier man kan bruke i Jupyter. Om du syns Jupyter ser ut som et godt verktøy og ønsker å jobbe litt med det anbefaler vi å prøve å merke seg noen av disse tastatursnarveiene. Du kan finne en liste med disse ved å gå på `Help -> Keyboard Shortcuts` .