

Git Guide - Commit yourself!

Nu har vi haft gang i git i noget tid, så alle burde have mindst en smule kendskab til git.

I den her guide tager jeg kun udgangspunkt i den terminal baseret version eftersom,

1. Jeg har kun brugt den terminal baserede version lige fra starten.

2. Når du lærer at bruge de forholdsvis simple commands, er det langt det hurtigste og smarteste at bruge.

3. Der er bare noget fedt ved at sidde at nørde i en terminal. "Det ser totalt hacker-agtigt ud".

From hub to local - Cloning a Repository

Som det første når man skal til at arbejde på projekt, som deles på et git repository. Er at få det klonet så man har det lokalt.



Dette gøres ved først at gå ind på repository-fanebladet:

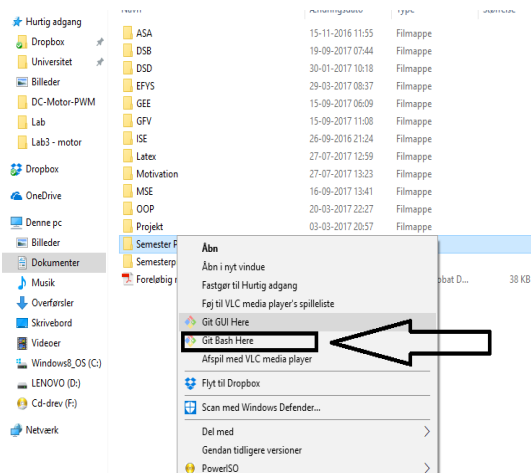
Og så kopiere linket som står i højre side:



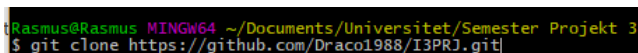
Lav nu en ny mappe det sted hvor du ønsker at projekt repo'et lokalt. Et forslag er at lave en mappe på C-drevet kaldet fx. Semesterprojekt 3 da det er her vi opbevarer vores filer.

Nå mappen er oprettet, højreklik nu på denne mappe og vælg:

Git Bash here:



Nu for du så git terminalen frem, og du kopierer nu repo-linket fra før ind i terminalen ved at trykke på Shift + Insert, så det står sådan her **git clone [her skrives repo linket som du kopierede før]**, og trykker så på enter. Så skulle det gerne se ca. sådan her ud:



```
Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3
$ git clone https://github.com/Draco1988/I3PRJ.git
```

Nu har du så repoet liggende lokalt og det du fremover laver, ligger kun lokalt indtil du har "committed" dine ændringer, og "pushed" dem til clouden.

Working with the repository - Basic Linux commands and little extra.

Nu kan vi så begynde at arbejde med repo'et. Du kan lave mapper, tilføje filer til mapperne, flytte rundt på filerne, og omdøbe dem derefter, præcis som du plejer igennem windows normale GUI. Men du kan også gøre det via terminalen, jeg vil vende til disse ting senere.

Men først tager vi de mest essentielle commands som i for brug for når i arbejder med git.

git pull

Når i starter på en ny arbejdsdag eller arbejder videre efter et par timers pause. Så start med at "pull" filerne fra clouden.

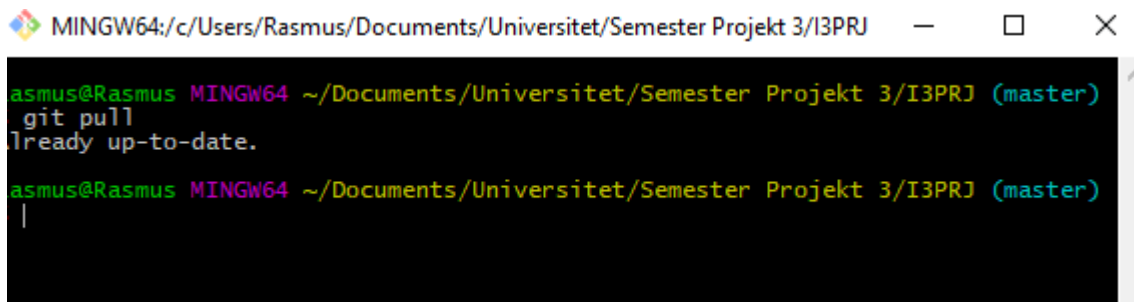
Det kan være der er filer essentielle for at du kan fortsætte med din opgave, eller der er lavet ændringer i forhold til en opgave.

Dette gør du ved commanden **git pull**, da det er et private repository bliver formentlig bedt om at skrive dit bruger navn og password.

Disse er de samme som dem du bruger på **github.com**. Nu du har indtastet disse skulle den gerne hente de nye/ændrede filer,

og merge det med de ting du har liggende, med mindre der ikke er nogen nye/ændrede filer så får du besked:

"already up to date". Nu er du så klar til at fortsætte arbejdet.



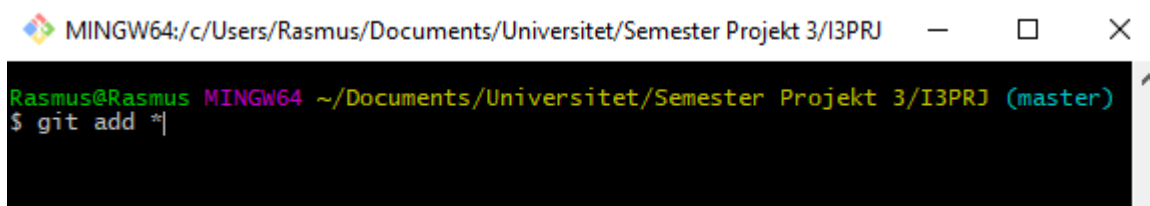
```
MINGW64:/c:/Users/Rasmus/Documents/Universitet/Semester Projekt 3/I3PRJ — □ ×
asmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ git pull
Already up-to-date.
asmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$
```

git add

Når du så har lavet et stykke arbejde og mener det er færdigt og ønsker at "pushe" det til repo'et i clouden.

Så skal du først fortælle git at den skal tilføje og "tracke" de filer du har tilføjet til repoet lokalt.

Det gør du ved brug af **git add**.



```
MINGW64:/c:/Users/Rasmus/Documents/Universitet/Semester Projekt 3/I3PRJ — □ ×
Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ git add *
```

Følgende kommandoer af git add burde kun benyttes, da det er så meget lettere.

git add *

Denne command "add'er" alle nye/ændrede filer, som står i den mappe som du i terminalen befinder dig i,

samt alle filer og mapper som ligger i underliggende mapper, fra din position.

Med mindre du har flyttet dig rundt, med change directory commanden så står du altid der hvor du "bashed" ind i git i starten.

Vi kommer tilbage til hvordan du bevæger der rundt i mapperne med git lidt senere.

git add -A

Er nemmere og mere overordnet MEN!, **Vigtigt!!! Jo nemmere det er, desto mindre har du også selv styr over hvad der egentlig sker**

Så det kræver at du selv holder styr på hvilke filer du har flyttet ind i diverse mapper osv. Så kort sagt den tilføjer ALT! nyt i jeres lokale repo.

Når vi kommer til change directory commanden så vil jeg anbefale i holder jer til **git add *** da i dermed bedre kan styre hvilke filer i tilføjer.

git reset *

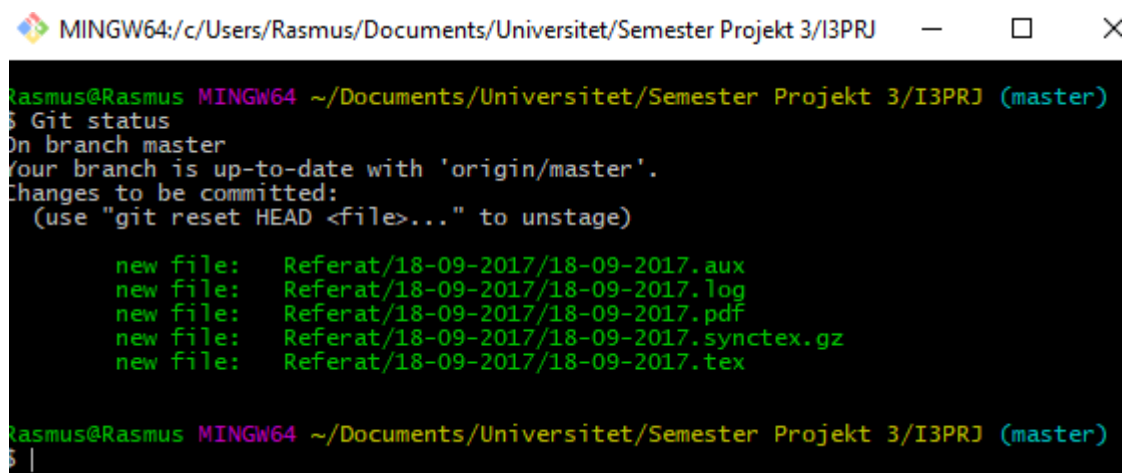
Hvis man kommer til at tilføje filer ved et uheld med **git add***, så kan man behændigt omgøre (undo) denne beslutning med **git reset ***.

Man må dog under ingen omstændigheder lave commit, før man laver reset! (da reset i så fald ingen virkning har)

git status

Når i har addet de filer i ønsker at gemme til repoet, så er det en god ide lige at tjekke om alt er som det skal være, er de rigtige filer blevet ændret/oprettet og tilføjet (add'et) til gits file tracker. Det gør i med commanden: **git status**

Så får i noget lignende det her:



```
MINGW64:/c/Users/Rasmus/Documents/Universitet/Semester Projekt 3/I3PRJ — □ ×
Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ Git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   Referat/18-09-2017/18-09-2017.aux
    new file:   Referat/18-09-2017/18-09-2017.log
    new file:   Referat/18-09-2017/18-09-2017.pdf
    new file:   Referat/18-09-2017/18-09-2017.synctex.gz
    new file:   Referat/18-09-2017/18-09-2017.tex

Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ |
```

git commit

Når i så mener i færdige med det i har lavet, og vil til at "pushe" det til clouden, så skal i først "commite" det i har lavet. Det er på sin vis den måde i fortæller git, "Jeg ønsker at gemme det jeg har lavet nu", så opretter git en ny virtuel version af projektet, medfølgende et unikt branch-id, så vis i laver ged i noget senere. Så kan vi gå tilbage til et tidligere stadie hvor der er mindre "ged" i projektet.

```
Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ Git commit -m "Referat fra sidste møde. #1"
[master 8cffb7b] Referat fra sidste møde. #1
5 files changed, 779 insertions(+)
create mode 100644 Referat/18-09-2017/18-09-2017.aux
create mode 100644 Referat/18-09-2017/18-09-2017.log
create mode 100644 Referat/18-09-2017/18-09-2017.pdf
create mode 100644 Referat/18-09-2017/18-09-2017.synctex.gz
create mode 100644 Referat/18-09-2017/18-09-2017.tex
Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ |
```

git push

Nu er i så klar til at smide det op på det online repo/clouden, det gøres med commanden: **git push**
I kan nu begynde at arbejde på projektet. HAVE FUN! You will cause git is fun!

```
$ git push
Counting objects: 8, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 42.77 KiB | 0 bytes/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To https://github.com/Draco1988/I3PRJ.git
  8b652d2..8cffb7b  master -> master
Rasmus@Rasmus MINGW64 ~/Documents/Universitet/Semester Projekt 3/I3PRJ (master)
$ |
```