Git how-to

Klaus Wogelius

17. september 2016

Indhold

Kapitel 1

GIT og GitHub

Git er et versionsstyringsværktøj.

1.1 Hent en arbejdskopi af et projekt i GitHub

1. I Windoze explorer (stifinder):
   (a) Gå derhen, hvor du vil have din arbejdskopi liggende
   (b) flyt pilen op til adressefeltet, højreklik og vælg "kopier adresse"
2. Start "Git Bash"
3. I Git Bash:
   (a) Skriv kommandoen "cd <paste den kopierede adresse>"
   (b) skriv "Git clone https://github.com/aqqalu/wireless-last-mile <enter>"
   (c) Gå ind i Arbejdsfolderen ("cd wireless-last-mile")

Herunder er et eksempel på proceduren:

```
kw@bar14825 MINGW64 /c/Users/kw/Desktop
$ cd /c/Users/kw/Documents

kw@bar14825 MINGW64 /c/Users/kw/Documents
$ git clone https://github.com/aqqalu/wireless-last-mile
Cloning into 'wireless-last-mile'...
remote: Counting objects: 18, done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 18 (delta 2), reused 18 (delta 2), pack-reused 0
Unpacking objects: 100% (18/18), done.

kw@bar14825 MINGW64 /c/Users/kw/Documents
$ cd wireless-last-mile/

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (master)
$
```

Når du har lavet en "clone"-kommando har Git automatisk gemt, hvorfra arbejdskopien kom. Du kan se dette med kommandoen "remote -v":

```
$ git remote -v
origin  https://github.com/aqqalu/wireless-last-mile (fetch)
origin  https://github.com/aqqalu/wireless-last-mile (push)

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (master)
$
```

1.2 Lav en opgave i arbejdskopien (Lav en "branch")

Når der skal laves en opgave i projektet, starter du med at lave en "branch" fra arbejdskopien med kommandoen "git branch <opgavenavn>".

Når jeg eksempelvis laver denne "Git-HowTo"-vejledning til vores projekt starter jeg med at lave en branch, som jeg vælger at kalde "git-howto":

```
$ git branch git-howto

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (master)
```

Derefter skiftes over til at arbejde i branch'en med opgaven ved hjælp af kommandoen "git checkout <opgavenavn>":

```
$ git checkout git-howto
Switched to branch 'git-howto'

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$
```

I denne branch kan så opgaven laves. Til eksempel har jeg kompileret LaTeX-dokumentet "git-howto.tex"med programmet "Tex-studio", efter at jeg i Git Bash havde lavet en checkout til branch'en "git-howto".

1.3 Lav et snapshot af dit færdige arbejde

Når du mener din opgave er færdig, eller hvis du ønsker at lave en sikkerhedskopi af opgaven gemmer du et snapshot af opgaven. Først ser du lige status med kommandoen "git status":

```
$ git status
On branch git-howto
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git checkout -- <file>..." to discard changes in working directory)

modified:   git-howto.aux
modified:   git-howto.pdf
modified:   git-howto.synctex.gz
modified:   git-howto.tex
modified:   git-howto.toc

no changes added to commit (use "git add" and/or "git commit -a")

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$
```

I nærværende kan man se at der er genereret ("modified") nogle filer i branch-folderen, fordi jeg kompilerede "git-howto.tex". Disse ændrede filer skal nu tilføjes til listen over filer, der skal medtages i snapshot'et:

```
$ git add git-how*
The following paths are ignored by one of your .gitignore files:
git-howto.log
Use -f if you really want to add them.

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$ git status
On branch git-howto
Changes to be committed:
(use "git reset HEAD <file>..." to unstage)

modified:   git-howto.aux
modified:   git-howto.pdf
modified:   git-howto.synctex.gz
modified:   git-howto.tex
modified:   git-howto.toc

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile(git-howto)
$
```

Slutteligt tager jeg ansvar for ændringerne med kommandoen "git commit -m <forklarende tekst..>" (en forklaring er obligatorisk):

```
$ git commit -m"git-howto dokument oprettet"
[git-howto 9830ac3] git-howto dokument oprettet
5 files changed, 42 insertions(+), 17 deletions(-)
rewrite git-howto.synctex.gz (100%)

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$ git status
On branch git-howto
nothing to commit, working tree clean

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$
```

Sådan. Snapshot'et er lavet og gemt for evigt :-)

1.4 Skaf overblik over hvilke branches der er i projektet

For at se, hvilke branches, der i øjeblikket er i projektet eller arbejdskopien kan anvendes kommandoen *"git branch"* uden parametre. Herunder ses resultatet af en sådan kommando. Det ses, at der i øjeblikket er 3 branches, "master", "git-howto" og "interessentanalyse". Stjernen ud for "git-howto" fortæller at jeg i øjeblikket er check'et ind i branch "git-howto":

```
$ git branch
```

```
git-checkout
* git-howto
interessentanalyse
master

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$
```

1.5 Skaf overblik over hvilke afgreninger der er i projektet

For at se, hvilke afgreninger, der i øjeblikket er i projektet eller arbejdskopien kan anvendes kommandoen *"git branch"* uden parametre. Herunder ses resultatet af en sådan kommando. Det ses, at der i øjeblikket er 3 afgreninger i arbejdskopien af projektet: "master", "git-howto" og "interessentanalyse". Stjernen ud for "git-howto" fortæller at jeg i øjeblikket er check'et ud på afgreningen "git-howto":

```
$ git branch
git-checkout
* git-howto
interessentanalyse
master

kw@bar14825 MINGW64 /c/Users/kw/Documents/wireless-last-mile (git-howto)
$
```

1.6 sammenkobl to grene

Fgreningen "interessentanalyse" blev lavet for at tilføje et afsnit i projektbeskrivelsen. Efter at dette afsnit er lavet ønsker jeg at sammenkoble afgreningen ind på mastersporet igen. Dette gøres med kommandoen "git merge". Proceduren er, at man går ind i det spor, som man vil koble afgreningen til og med kommandoen "git merge" kobler afgreningen på:

1.7