

Workshop git demo

Gilles Coremans

21 November 2018

```
# This work is licensed under a Creative Commons Attribution –  
# NonCommercial–ShareAlike 4.0 International license.
```

```
# Door Gilles Coremans voor ULYSSIS
```

```
## Installatie ##
```

```
git config —global user.email "email@example.org"  
git config —global user.name "Uw Naam"  
git config —global core.editor nano  
git config —global init.defaultBranch main
```

```
Verifieer installatie: git -v of git —version  
Verifieer config: git config —get user.email
```

```
Gitlab/hub = servers
```

```
## Een eerste commit ##
```

```
ls # in basedir
```

```
mkdir repo
```

```
ls
```

```
cd repo
```

```
ls
```

```
git init # in basedir/repo  
# Vermijd spaties in path
```

```
ls
```

```
ls -A # toont hidden files (.git)
```

```
git status
```

```
nano boodschappen.txt # Nieuwe file
```

```
ls
```

```
git status
```

```
git add boodschappen.txt
```

```
ls
```

```
git status
```

```
git commit
```

```
ls
```

```
git status
```

```
## Wat hebben we gedaan? ##
```

```
git log # q om uit de pager te gaan!!!
```

```
git show # -> specifiek voor bepaalde commit
```

```

## git add ##
# staged vs working tree , en ook git add -u
ls
git status
nano boodschappen.txt          # pas iets aan
ls
git status
git add -u
ls
git status
nano boodschappen.txt          # pas iets aan
ls
git status                      # zowel unstaged changes als in de index
git commit
ls
git status
git log
git show
nano boodschappen.txt          # Pas niks aan, note changes tov HEAD

# git add -u add geen untracked files
ls
git status
nano dinsdag.txt                # Nieuwe (untracked) file
ls
git status
git add -u
ls
git status
git add dinsdag.txt
git status
git commit
ls
git status
git log
git show

# git restore --staged <file >
ls
git status
nano dinsdag.txt
nano boodschappen.txt
git status
git add -u                      # Add alles
git status
git restore --staged dinsdag.txt # Deze willen we niet
git status                      # Verdwenen uit staging area
git commit

```

```

git status                                # Changes in dinsdag.txt zijn er nog steeds
git add dinsdag.txt
git commit
git status
git log

# .gitignore
nano niet-tracken.txt                    # Nieuwe file! Zonder spatie!
git status
nano .gitignore                          # Voeg "niet-tracken.txt" toe
git status                                # niet-tracken.txt staat er niet meer!
ls                                         # maar bestaat nog wel
git add niet-tracken.txt                 # git waarschuwt zelfs
git status
git add .gitignore                       # .gitignore kan mee in repository
git status
git commit
git show
ls                                         # "niet-tracken.txt" bestaat
git status                                # Maar staat niet als "untracked"!
rm niet-tracken.txt                      # Niet meer nodig

### git commit ###
# git commit -m
ls
git status
nano boodschappen.txt                    # pas iets aan, simpele change voor -m
git status
git add -u
git status
git commit -m "Voeg een boodschap toe"
git status
git show

# git commit -a
nano dinsdag.txt                          # pas iets aan
git status
git commit -a                            # toont wat er gecommited zal worden
git status
git show

# git commit -a -m
nano boodschappen.txt                    # pas iets aan
nano dinsdag.txt                          # pas iets aan
git status
git commit -a -m "Nog meer boodschappen"
git status
git show

```

```

## Commits bekijken ##
# git show
git show
git log
git show (hash van HEAD)
git show main # ← alledrie hetzelfde!

git show HEAD~1 # HEAD~1 = relatieve verwijzing
git log
git show (random andere hash)

# git log
git log
git log --oneline

# git diff
nano boodschappen.txt # verander iets
nano dinsdag.txt # verander iets
git add boodschappen.txt # enkel boodschappen.txt!
git status
git diff
git diff --staged
git diff HEAD~1 HEAD
git diff HEAD~3 HEAD
git commit -a # zorg dat working directory clean is voor volg

PAUZE

## Remotes ##
# Repositories aanmaken
cd .. # in basedir
git clone repo --bare upstream # Clone naar een bare "upstream" repository
ls
git clone upstream second # Clone een tweede repository om van te pullen
cd second # in basedir/second
ls
git status
cd .. # in basedir
ls
cd repo # in basedir/repo
git remote add origin ../upstream
ls

# git push/pull
git push
git push --set-upstream origin main
nano boodschappen.txt # pas iets aan
git commit -am "Commit message"
git log
git push

```

```

cd ../second # in basedir/second
git log
git fetch
git log --all
nano boodschappen.txt # pas niks aan
git pull
git log
nano boodschappen.txt # pas niks aan
cd ../repo # in basedir/repo

## Branches ## # ← setup voor merge!
# git branch
git status
git branch # branches tonen
git branch nieuw
git status
git switch nieuw
git status
git log
nano dinsdag.txt # Verander iets
git commit -a
git log # note twee branches, note ook dat HEAD → nieuw

git switch main
git status
ls
git show
nano dinsdag.txt # verander niks, maar note de veranderingen
nano boodschappen.txt # verander iets
git commit -a
git status
git log --all --oneline --graph # 2 verschillende branches!

# git switch -c (en setup voor merge)
git status # op main!
git switch -c feature
git status
nano boodschappen.txt # verander iets
git status
git add -u
git commit
git log --all --oneline --graph

# Note: in de branch "nieuw" is enkel dinsdag.txt veranderd en in main niet, du
## Merging ##
git switch main
ls
git status
git show
git log --oneline --graph --all

```

```

# fast-forward
git merge feature
ls
git status
nano boodschappen.txt          # verander niks
git show
git log --oneline --graph --all

# merge
git merge nieuw
ls
git status
nano boodschappen.txt          # verander niks
nano dinsdag.txt
git log --oneline --graph --all
git show

### Conflicts ###
git switch -c dev
nano boodschappen.txt          # Verander een regel , note welke
git commit -a
ls
git status
git log --oneline --graph --all

git switch main
ls
git status
nano boodschappen.txt          # verander dezelfde regel opnieuw
git commit -a
ls
git status
git log --oneline --graph --all

git diff main dev
git merge dev                    # conflict !!!
ls
git status
nano boodschappen.txt
git status                      # nog steeds binnen merge commit
git add boodschappen.txt
git merge --continue
ls
git status
git show
git log --oneline --graph --all

```

```
## git reset ##
# git reset --hard: slechte change
rm boodschappen.txt # verwijder bestand uit git, rm en add
ls
git status
git commit
ls
git status
git log --oneline --graph
git reset --hard HEAD~1
git log --oneline --graph --all
git status
ls
nano boodschappen.txt
```