

Git voor beginners

Jonathan Valgaeren

UL  **SSIS**

23 November 2023

This work is licensed under a Creative Commons "Attribution-NonCommercial-ShareAlike 4.0 International" license.



Gemaakt door Gilles Coremans

Installatie



Windows <https://www.git-scm.com> en Git Bash

Mac <https://www.git-scm.com> en Terminal

Linux Package manager en uw favoriete terminal

- ▶ `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`

Wat is git?

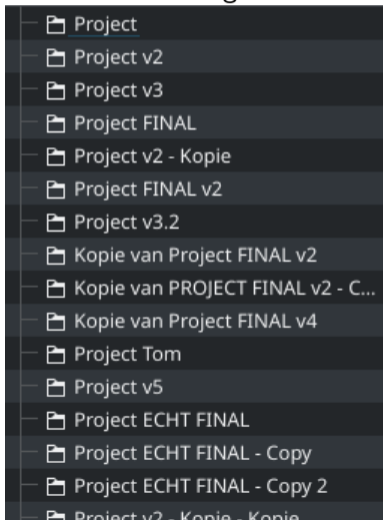


- ▶ Versiecontrolesysteem
- ▶ Door Linus Torvalds (auteur van Linux!)
- ▶ Vanwege onvrede met bestaande systemen:
 - ▶ Git is sneller
 - ▶ Git is gedistribueerd
 - ▶ Git is heel krachtig
- ▶ NIET GitLab of GitHub!

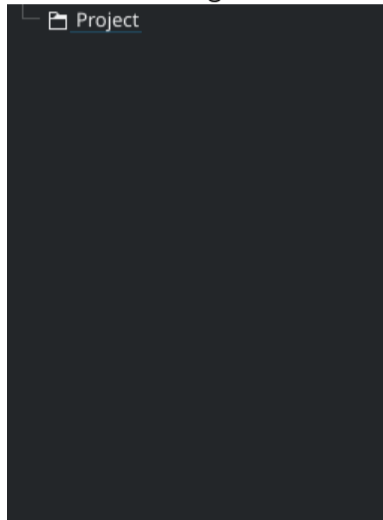
Waarom git?



Zonder git:



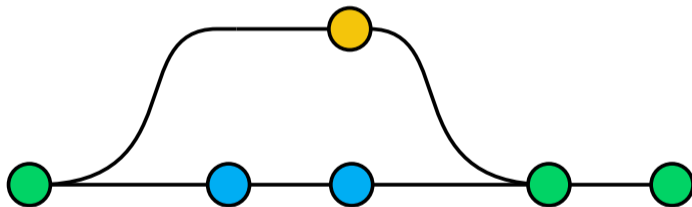
Met git:



Naamgeving



- ▶ Een project in git is een *repository*
- ▶ Veranderingen groeperen in een *commit*
- ▶ Commits volgen elkaar op in een boom
- ▶ Een tak van de boom is een *branch*
- ▶ Een branch kan op elk punt ontstaan of samenkomen



Naamgeving



Een file in een repository kan zich in vier verschillende “staten” bevinden:

Untracked Git volgt deze file (nog) niet

Modified De file is veranderd sinds de laatste versie

Staged De veranderingen in de file zijn klaar voor de *volgende* versie (commit)

Unmodified De file is hetzelfde als in de laatste versie (als in HEAD)

▶ `git status` toont in welke staat elke file zit!

Een eerste commit



- ▶ Tip voor we beginnen: `git help`
- ▶ Repository maken: `git init`
 - ▶ Nieuwe lege repository
 - ▶ Enkel lokaal
- ▶ `git add <file>`: Files toevoegen aan de staging area (volgende commit)
- ▶ `git commit`: Alle veranderingen in de staging area groeperen in nieuwe *commit* (versie) die volgt op HEAD (vorige versie)

Wat hebben we gedaan?



- ▶ `git log`: Alle commits tonen (wie & wanneer)
- ▶ `git show`: Veranderingen in laatste commit bekijken

git add



- ▶ Ook mappen als argument
- ▶ `git add -u`: Voeg elk **tracked** bestand toe aan de staging area.
- ▶ `git restore --staged <file>`: Haal `<file>` uit de staging area
- ▶ `.gitignore`:
 - ▶ Bestand, meestal in de bovenste map van de repository
 - ▶ Vertelt git welke bestanden en mappen te negeren
 - ▶ Ondersteunt `*` als wildcard
 - ▶ Makkelijk te maken op <https://gitignore.io>

git commit



- ▶ `git commit -m "Commit message"`: Voeg inline een commit message toe
- ▶ `git commit -a`: Voeg alle **tracked** files toe aan de staging area, en maak dan een nieuwe commit
- ▶ `git commit -a -m "Commit message"`: Combinatie van `-a` en `-m`!

Commits bekijken



- ▶ `git show <commit>`: Details en veranderingen van één commit
- ▶ `git log <commit>`: Geschiedenis van de commit
 - ▶ `git log --oneline`: Eén regel per commit
- ▶ `git diff`: Veranderingen tussen staging area en modified bestanden
 - ▶ `git diff --staged`: Veranderingen tussen staging area en HEAD
 - ▶ `git diff <commit1> <commit2>`: Veranderingen in <commit2> ten opzichte van <commit1>

Inleiding
○○○

Introductie tot git
○○○○

De basis van git
○○○●

Branches en remotes
○○○○○○○

Een diepere kijk in git
○○○

Pauze

Remotes



- ▶ Tot nu toe: alles lokaal
- ▶ Wat is een remote?
 - ▶ een repository
 - ▶ gehost op internet of ander netwerk
 - ▶ samenwerken aan project
- ▶ Hoe instellen?
 - ▶ `git remote add origin <url>`: Handmatig URL toevoegen
 - ▶ `git clone <url>`: Download een repository, git stelt remotes automatisch in

Pushen en pullen



- ▶ `git fetch`: Haal veranderingen van remote, maar voer ze niet door
- ▶ `git pull`: Veranderingen van remote ophalen en direct doorvoeren
- ▶ `git push`: Veranderingen pushen naar remote

Branches



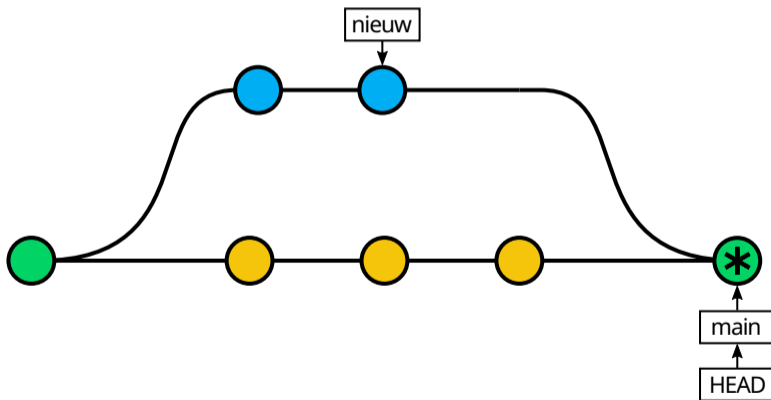
- ▶ Maak aanpassingen zonder main branch te beïnvloeden
- ▶ Branches zijn verwijzingen naar een commit (en HEAD wijst gewoon naar de huidige branch)
- ▶ `git branch`: Branches beheren
- ▶ `git switch <branch>`: Wissel naar de branch of commit <branch>
- ▶ `git switch -c <branch>`: Hetzelfde als `git branch <branch>` en dan `git switch <branch>`: maak een nieuwe branch en wissel ernaar
- ▶ `git log --all`: Toon de geschiedenis van **alle** branches
- ▶ `git log --graph`: Toon de verbindingen tussen commits

Merging



- ▶ `git merge <branch>`: Merget een branch in de huidige branch
- ▶ `git pull` is gewoon `git fetch` gevolgd door `git merge`!
- ▶ `<branch>` is een voorouder van de huidige branch: doe niks (we hebben alle veranderingen al!)
- ▶ `<branch>` ligt in het verlengde van de huidige branch: `fast-forward`
- ▶ `<branch>` is eerder afgetakt: maak een *merge-commit* met twee ouders: één op elke branch

Merging: grafisch



Conflicts



- ▶ Veranderingen in hetzelfde bestand worden slim samengevoegd
- ▶ Veranderingen op zelfde regel → conflict!
- ▶ Voordat de merge gefinaliseerd wordt: conflict oplossen
- ▶ Manueel of met een merge tool: `git mergetool`
- ▶ Conflicts opgelost: `git merge --continue`
- ▶ Merge aflasten: `git merge --abort`

git reset



- ▶ `git reset` laat je teruggaan naar een vorige commit
- ▶ `git reset <commit>`: Update de huidige branch *en* de staging area naar de gegeven commit, maar laat de working tree met rust
- ▶ `git reset --hard <commit>`: Update de huidige branch, de staging area *en* de working tree naar de gegeven commit
- ▶ Changes uit de staging area verwijderen, teruggaan naar de huidige commit, laatste commit wissen...
- ▶ Verplaatst ook de branch pointer = geschiedenis herschrijven!
- ▶ Commit wissen = geschiedenis herschrijven!

git stash



- ▶ Veranderingen in je working directory tijdelijk opslaan, buiten de tree
- ▶ `git stash push`: Sla uncommitted veranderingen tijdelijk op en reset working tree
- ▶ `git stash pop`: Voeg changes toe en leeg stash

Making amends



- ▶ `git commit --amend`: Voeg changes in de staging area toe aan *vorige* commit
- ▶ **Herschrijft geschiedenis!**
- ▶ Nuttig voor foute commits die nog niet gepusht zijn

En verder?



- ▶ `man git`: ingebouwd handboek!
- ▶ <https://git-scm.com/book>: zeer volledig boek over Git
- ▶ StackOverflow: praktische oplossingen en uitleg