GitItDone

Slides voor GitItDone workshop van het IT-lab


Git logo

Meevolgen op:

https://hogent-it-lab.github.io/gititdone-workshop/slides

Wat is Git?

- Git is een version control system (VCS)
  - Hou een historie bij van je project en alle wijzigingen
  - Wijzigingen worden systematisch bijgehouden
  - Keer eenvoudig terug naar oudere versies (handig bij fouten...)
  - Leuk om evolutie en voortgang van project te zien

Een beetje geschiedenis...

- Ontworpen en uitgebracht in 2005
- Gemaakt door Linus Torvalds voor development van Linux
- Grote populariteit en meestgebruikte VCS vandaag de dag

Account maken op een git server

https://www.github.com (**GitHub ≠ Git**)


git_signup

Je git account

Stuur nu het gebruikte email-adres door naar
alexander.veldeman@hogent.be

Git installeren

- Beschikbaar voor Windows, MacOS en Linux (uiteraard)
  - Download voor jouw systeem de juiste versie
  - Windows: Git Bash is ook een optie

Installatie controleren

Commando `git`
Bijvoorbeeld: `git --version`

```
C:\Users\sion_\Documents\gititdone-workshop> git --version
git version 2.30.1.windows.1
```

Git configureren

Commando: `git config`

- `git config -l`: oplijsten
- `git config --global -l`: instellingen voor heel het systeem
- In een git map (later): `git config --local -l`
- Instellingen aanpassen: `git config --global user.name NieuweNaam`

- Email en naam instellen **op waarden van GitHub account**

```
git config --global user.name <USERNAME>
```

```
git config --global user.email <JOUW@EMAIL.ADRES>
```

Resultaat:

```
git config --global -l
user.email=sion.verschraege@hogent.be
user.name=sionverschraege
```

Geavanceerde configuratie

- Lokaal heeft voorrang op globaal
- Mensen die meerdere accounts hebben:
  - Globaal meest gebruikte naam/email
  - Lokaal per repo voor andere account
  - Verschillende SSH-keys nodig

Git configureren - SSH sleutels

Data **encrypteren** en **handtekenen**
Hoe genereer ik zo'n sleutelpaar?

```
ssh-keygen -t ed25519 -C "email@hogent.be"
```

- aanhalingstekens (`"`) moeten er staan

Terug te vinden in `[je home folder]/.ssh`:
 keys

Public key uploaden op GitHub

Settings -> SSH and GPG keys -> new SSH key

- Kies een goede naam
- Kies "Authentication Type"
- Plak de sleutel **vanuit het pub bestand** in het tekstvak
- Test met `ssh -T git@github.com`

```
ssh -T git@github.com
Hi sionverschraege! You've successfully authenticated, but GitHub does not provide shell access.
```

Repository clonen

Code van op de **git server** (github, **remote repository**) kopiëren naar eigen pc (**local repository**)

- Deze slides (gemaakt in **markdown-code**) zijn publiek!
- `git clone git@github.com:HOGENT-IT-Lab/gititdone-workshop.git`


Git clone link

Git - commits

- Om veranderingen bij te houden, maken we gebruik van **commits**
- Commits bekijken: `git log` in git-mapje
- Commits bevatten één of meerdere **wijzigingen** in onze repository
- Commits hebben altijd een bijhorende **commit message**
- Commits hebben altijd een unieke **commit hash**

Commits - tips en best practices

- Probeer je commits **atomair** te houden! (commit heeft een doel)
- Schrijf duidelijke commit messages!
- Kom tot overeenkomst/structuur met jouw team (samenwerken)

Wat is een .gitignore bestand?

Soms wil je sommige zaken niet bijhouden in jouw version control (denk aan wachtwoorden, sensitieve data,...)

- `.gitignore` definieert wat je niet wilt tracken
- Er bestaan heel wat templates voor verschillende projecten!
- Kan ook wildcards gebruiken

.gitignore - voorbeeld Java project

```
# Compiled class file
*.class

# Log file
*.log

# Package Files
*.jar
*.war
*.nar
*.ear
*.zip
*.tar.gz
*.rar
```

Git workflow

Git - command line versus GUI

Git kan je zowel via de CLI als GUI gebruiken

- CLI: snel en efficiënt, iets hogere learning curve (maar niet veel!)
- GUI: handig voor een visueel overzicht, maar er gebeurt veel *under the hood*
- Vele IDE's hebben Git support ingebouwd!

Voorbeeld: GitHub Desktop

Basiscommando's - algemeen

- Huidige toestand/situatie bekijken

```
git status
```

Gebruik dit commando na/voor elk ander commando!

- Historie/commits weergeven

```
git log
```

Basiscommando's - nieuwe repo

- Van een lokale map een git repository maken

```
git init
```

- Een repository binnenhalen (klonen)

```
git clone <URL>
```

Basiscommando's - lokaal naar remote

- Wijzigingen in working directory aan staging toevoegen

```
git add <FILE>
```

- Alles toevoegen aan staging

```
git add .
```

- Wijzigingen in stage aan de lokale git-repository toevoegen

```
git commit
```

- Meteen een (duidelijke!) commit message toevoegen

```
git commit -m "DIT IS MIJN COMMIT MESSAGE"
```

- Lokale repository naar een remote repository

```
git push
```

Tip: stel pushen niet uit! Vermijd merge conflicten (zie later)

Deze slide is een herinnering: Alexander moet tijdens de volgende slide een testfiletje maken en pushen naar HOGENT-IT-Lab/gititdone-workshop.git

Tijd voor een testpush!

- Maak een repository op github.com
- Clone: `git clone git@github.com:sionverschraege/testrepo.git`
  - Clone NIET in je andere repo!
- Maak een nieuw bestandje aan **in de lokale working directory**
- Voeg het bestand toe aan de **staging area**: `git add test.txt`
- Maak een **commit** van alles in de staging area:
  `git commit -m "added test file"`
- **Push** naar de remote repo: `git push`
- Bewonder je bestand op github.com

Basiscommando's - remote naar lokaal

- Remote repository naar lokale repository

```
git fetch
```

- Dit verandert je *working copy* niet, het updatet enkel de data in je **lokale repository**
- Je moet een remote repository ingesteld hebben (zie verder)
  - Dit gebeurt automatisch bij clonen, niet bij init

Basiscommando's - remote naar lokaal

- Lokale repository naar working directory

```
git merge
```

- Tegelijk fetchen en mergen

```
git pull
```

Tijd voor een testpull!

- Vraag Alexander of het testfiletje aangemaakt is
- Ga terug naar de lokale map voor `HOGENT-IT-Lab/gititdone-workshop.git`
- Gebruik `git fetch` om de nieuwe informatie op te vragen
- Gebruik `git merge` om het bestand in je workspace te krijgen
- Bewonder het bestand in je lokale mapje

Basiscommando's - oeps...

- **Unstagen** van wijziging

```
git reset
```

- Recentste commit (lokaal) ongedaan maken

```
git reset HEAD~1
```

- Commit reverten met nieuwe commit

```
git revert
```

Basiscommando's - remote

- Checken of er een remote is ingesteld

```
git remote
```

- Voor meer informatie (URL, push/fetch)

```
git remote -v
```

- Remote toevoegen aan lokale repository (bv origin)

```
git remote add origin <URL>
```

Order matters...

Git - merge conflicten

- Wat zijn merge conflicten?
- Wanneer ontstaan deze?
- Hoe kan je deze oplossen?

Al dit en meer in de volgende live demo

Doe-het-zelf merge conflict

- Clone het zandbakproject
  - `git clone git@github.com:HOGENT-IT-Lab/gititdone-zandbak.git`
- Spreek met je buur of buren af welke bestandsnaam je gebruikt
- Maak elk, lokaal, dit bestand aan, en zet er elk iets (anders) in
  - Hou het alstublieft een beetje deftig
  - Vergeet niet dat commits op naam staan
- Push allemaal dit bestand
- Los de merge conflicten op
- Pull, en bewonder de samengestelde bestandjes

Git - branches

- "Maar ik wil niet voor elke commit merge conflicts oplossen!"
- "Maar ik wil een versie van mijn software afsplitsen!"
- "Maar ik wil mijn code niet officieel maken vóór alles werkt!"

Git - branches

- Oplossing: **branches**
- Branches zijn **parallelle versies**
- Branches kunnen later weer **samengevoegd** worden

Branches - commando's

- Je wil vertrekken van commit X. Ga naar die commit met

```
git checkout <COMMIT_HASH>
```

- Dit mag een kortere hash zijn
- Nieuwe branch aanmaken

```
git branch <NAAMBRANCH>
```

Branches - commando's

- Wisselen naar branch

```
git checkout <NAAMBRANCH>
```

- Maak een commit op die branch

```
git add <BESTAND>
git commit -m "<COMMIT MESSAGE>"
```

Branches - doe het zelf

- Zoek in de zandbak-repo je eigen commit
- Maak een branch vertrekkende van die commit
  - Gebruik een goede en unieke naam!
- Verander je eigen file een beetje en push naar de nieuwe branch
- Bewonder alle branches op github

GitHub specifiek

- Heel wat specifieke mogelijkheden binnen het platform
- GitHub Actions: automatisatie, testen en meer
- GitHub Pages: host statische websites en projecten gratis
- Webhooks: stel meldingen in voor veranderingen in repository (Discord, Slack, mail, ...)
- Pull requests: review en bekijk code/wijzigingen voor een push

GitHub Pages

- Manier om statische websites te hosten
- Use case: deze slides! (in combinatie met GitHub Actions)
- Eenvoudige websites, basic applicaties, ...

GitHub Actions

- Automatisatie en workflows (pipelines, testen, software bouwen,...)
- Zelf instellen van workflows en heel veel customisation

GitHub Actions - voorbeeld

```
name: "Export and publish slides"
on:
  workflow_dispatch:

jobs:
  publish:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout repository
        uses: actions/checkout@v4

      - name: Convert markdown to PDF
        uses: KoharaKazuya/marp-cli-action@v3
```

Git rebase

- "Ik wou dat mijn branch begon op een andere commit!"
- "Ik wou dat mijn commit verderbouwde op een andere commit!"
- Oplossing: aanpassen van de **parent** van een commit of branch met `git rebase`

```
git rebase main newBranch
```

Handige/interessante links

- Git(Hub) guide van HOGENT - https://hogenttin.github.io/git-hogent-gids/
- Leren branchen - https://learngitbranching.js.org/
- Templates .gitignore - https://github.com/github/gitignore
- Wat als het allemaal in de soep loopt? - https://ohshitgit.com/