

# Git en versiebeheer

niet alleen voor coders

Dawid Zalewski

April 14, 2019

# Intro

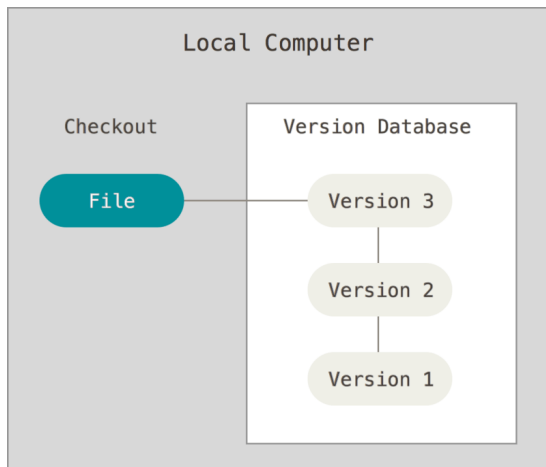
# Versiebeheersysteem

Versiebeheer is het systeem waarin veranderingen in een bestand of groep van bestanden over de tijd wordt bijgehouden, zodat je later specifieke versies kan opvragen.

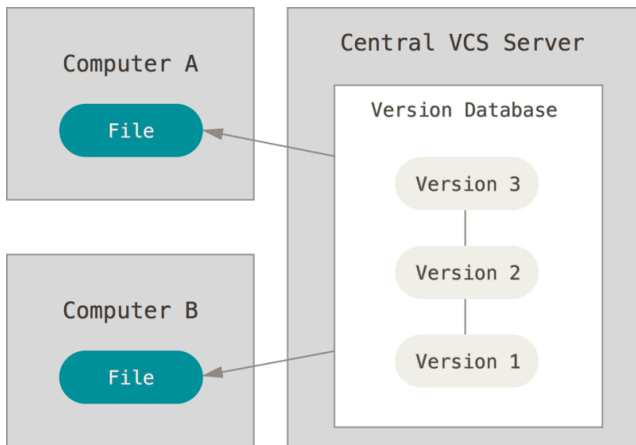
# Waarom versiebeheer

- Terughalen van eerdere versies van bestanden of het hele project
- Bekijken wijzigingen tussen twee momenten in de tijd
- Opvolgen wie wat aangepast heeft

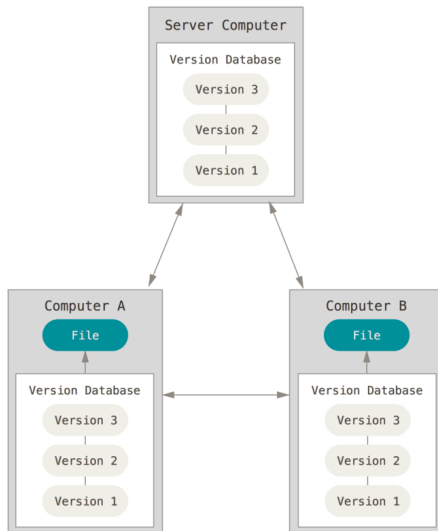
# Lokale versiebeheersystemen



# Gecentraliseerde versiebeheersystemen



# Gedistribueerde versiebeheersystemen



# Wat is git

Git is een *gedistribueerd* versiebeheersysteem

- Elk teamlid kan onafhankelijk werken
- Bijna alle handelingen zijn **lokaal**
- Iedereen heeft een lokale kopie van de database (repository)



# Om te beginnen

- Git client: <http://git-scm.com/download/>
  - Pak de juiste versie voor jouw systeem,
  - Inclusief *Git Bash* (alleen Windows)
- GUI tool: <http://git-scm.com/downloads/guis>
  - Bijv. [Github Desktop](#)

## Info

Wij gaan (proberen) geen GUI tools te gebruiken.

# Aan de slag - een paar instellingen

Git moet weten wie jij bent:

```
$ git config --global user.name "Dawid Zalewski"
```

```
$ git config --global user.email "d.r.zalewski@saxion.nl"
```

# Aan de slag - tekst editor

Je kan ook de default tekst editor aanpassen:

Voor Notepad++:

```
$ git config --global core.editor  
"'notepad++.exe' -multiInst -nosession"
```

For Visual Studio Code:

```
$ git config --global core.editor "code --wait"
```

# Aan de slag - controleer de instellingen

```
$ git config --list
```

## Lokaal versiebeheer

# Initialiseer repository

```
$ mkdir my_project
```

```
$ cd my_project/
```

```
$ git init
```

```
Initialized empty Git repository in ...
```

```
$ git status
```

```
On branch master
```

```
Initial commit
```

```
nothing to commit (create/copy files  
and use "git add" to track)
```

# Drie toestanden

Er zijn drie toestanden waarin bestanden zich kunnen bevinden:

- *gewijzigd* ('modified'),
- *voorbereid* voor een commit ('staged')
- *gecommit* ('committed'),

# Drie toestanden

## Modified

Een bestand is gewijzigd maar nog niet naar de database gecommit

## Staged

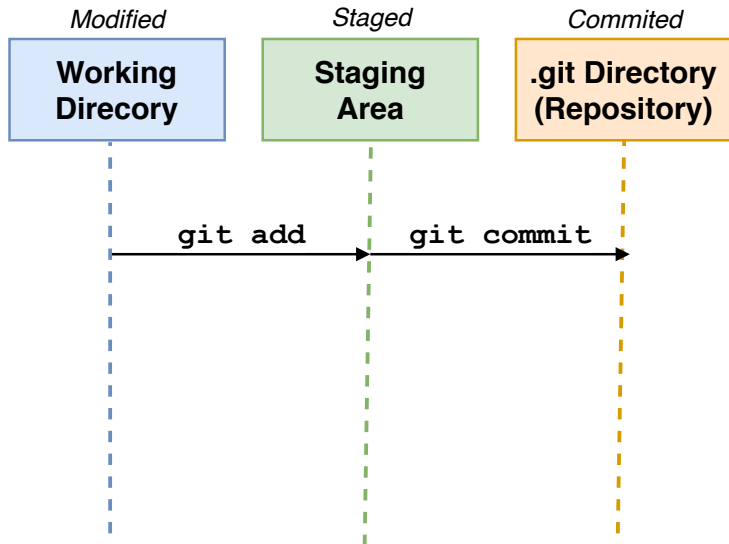
Een aangepast bestand wordt in de volgende commit meegenomen

## Committed

Alle data zit veilig in de lokale database



## Git workflow & omgeving



# Werkomgeving

## Working directory

De directorystructuur en bestanden waarin je wijzigingen aanbrengt

## Staging Area

*Tussenstation* tussen **working directory** en **repository**

Laat toe wijzigingen selectief te *committen*

## Repository

Verzameling (backup database) van alle commits, branches, tags, ...

# Eerste commit

Maak een nieuw bestand aan (readme.txt) in de `my_project` map

Voor VS Code gebruikers:

```
$ code readme.txt
```

- Type wat tekst in en sla op.

# Contoleer

```
$ ls -l
```

```
total 1
```

```
-rw-r--r-- 1 zalda 197609 7 Apr 11 12:55 readme.txt
```

# Git status

```
$ git status
```

```
On branch master
```

```
No commits yet
```

```
Untracked files:
```

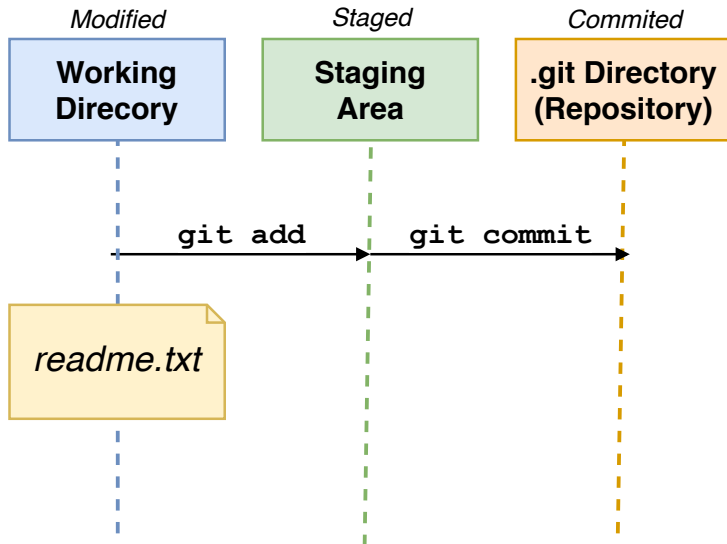
```
(use "git add <file>..." to include in what  
will be committed)
```

```
    readme.txt
```

```
nothing added to commit but untracked files present
```

```
(use "git add" to track)
```

## Waar is `readme.txt`



## Naar de *Staging Area*: `git add`

```
$ git add readme.txt
```

```
$ git status
```

```
On branch master
```

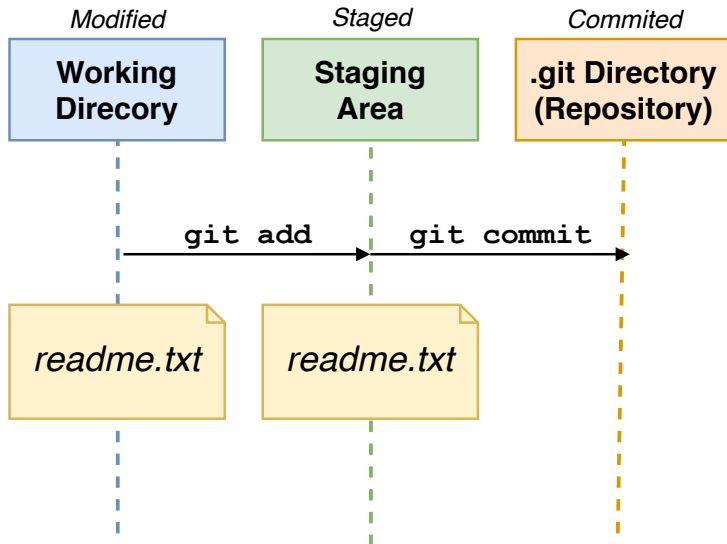
```
Initial commit
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file:   readme.txt
```

## Waar is `readme.txt`



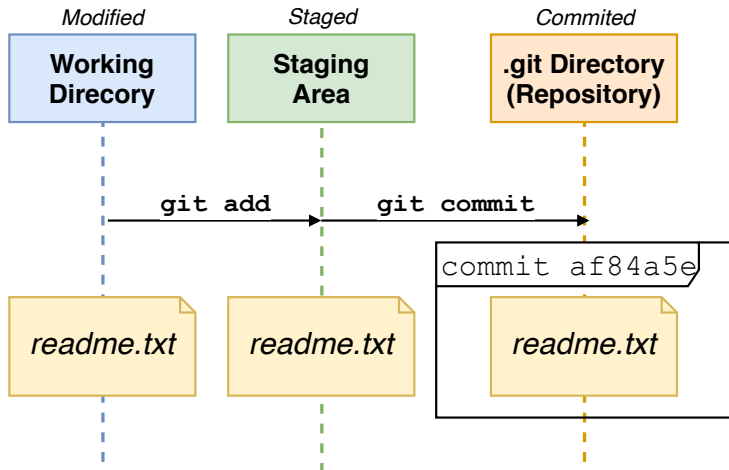


## Een commit toevoegen: `git commit`

```
$ git commit -m "readme.txt toegevoegd"
[master (root-commit) af84a5e] readme.txt toegevoegd
 1 file changed, 1 insertion(+)
 create mode 100644 readme.txt

$ git status
On branch master
nothing to commit, working tree clean
```

# Waar is readme.txt



## Even checken

```
$ git log
commit af84a5e... (HEAD -> master)
Author: Dawid Zalewski <d.r.zalewski@saxion.nl>
Date: Thu Apr 11 14:56:28 2019 +0200
```

readme.txt toegevoegd

Het originele bestand blijft in de map zitten.

Trouwens ook in de Staging Area.

## Laten we iets aanpassen

Open `readme.txt` en voeg wat tekst toe / pas iets aan.

```
$ git status
```

```
On branch master
```

```
Changes not staged for commit:
```

```
(use "git add <file>..." to update what  
will be committed)
```

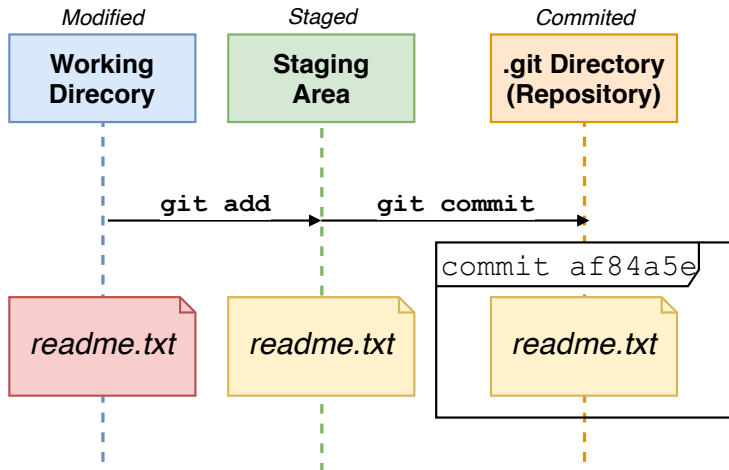
```
(use "git checkout -- <file>..." to  
discard changes in working directory)
```

```
        modified:   readme.txt
```

```
no changes added to commit
```

```
(use "git add" and/or "git commit -a")
```

# Git herkent de wijzigingen



# Alweer naar de Staging Area

```
$ git add .
```

```
$ git status
```

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

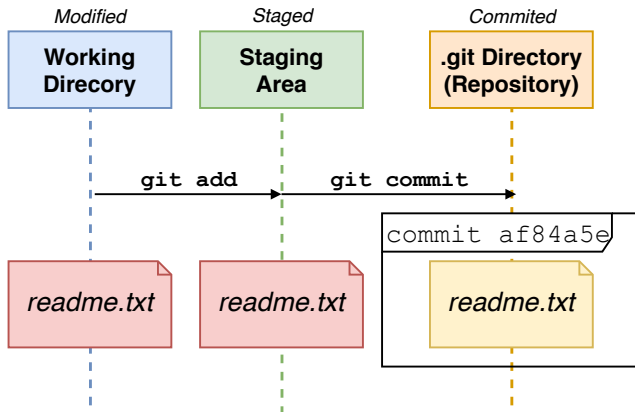
modified: readme.txt

```
git add .
```

```
$ git add .
```

- `.` is de huidige directory (incl. onderliggende)
- Je kan ook bestanden individueel toevoegen (door zijn namen te vermelden)
- **Let op!** Werkt enkel *binnen* de directory met repository

## Staging area na git add .



Let op: `readme.txt` zit nog steeds in de map.

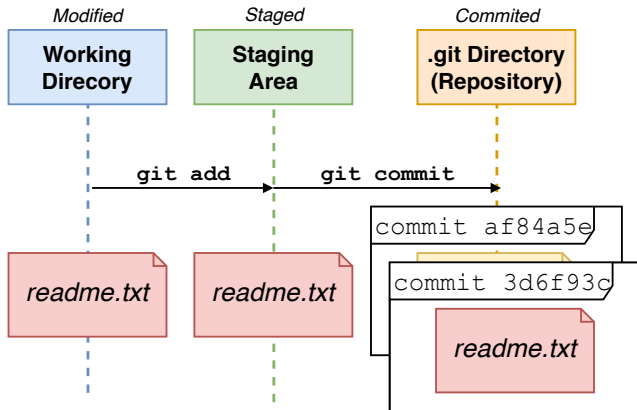
git erkent slechts dat hij ook voor het commiten klaar is.



# Wijzigingen commiten

```
$ git commit -m "een regel in readme.txt toegevoegd"
[master 3d6f93c] een regel in readme.txt toegevoegd
1 file changed, 1 insertions(+), 0 deletion(-)
```

## De situatie na de commit



Wij hebben nu twee commits!

## Een bestandje verder (zelf proberen)

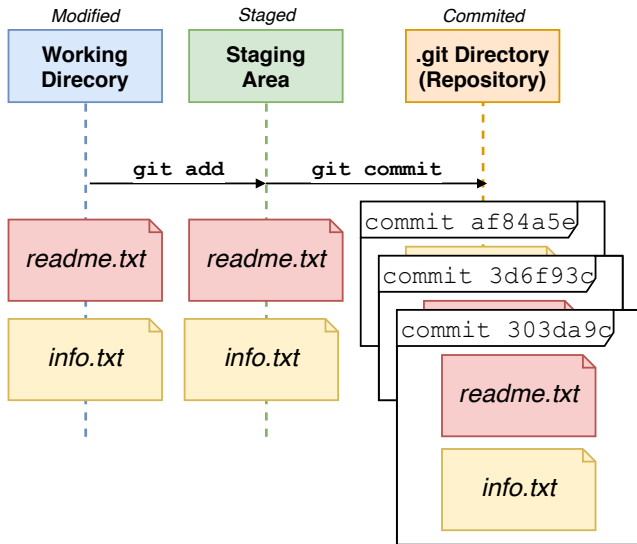
Maak een nieuwe file aan (bijv. info.txt) en:

- Voeg hem aan de Staging Area toe.
- Commit hem.

## Een bestandje verder (code)

```
$ touch info.txt
$ code info.txt
$ git add .
$ git commit -m "Initiële revisie info.txt"
$ git log --oneline
```

## Nu hebben wij twee files



# Git time machine

## Locale wijzigingen ongedaan maken

Er zitten wijzigingen in een bestand (bijv. `info.txt`) die je ongedaan wilt maken.

```
$ git status
```

On branch master

Changes not staged for commit:

```
(use "git add <file>..."
```

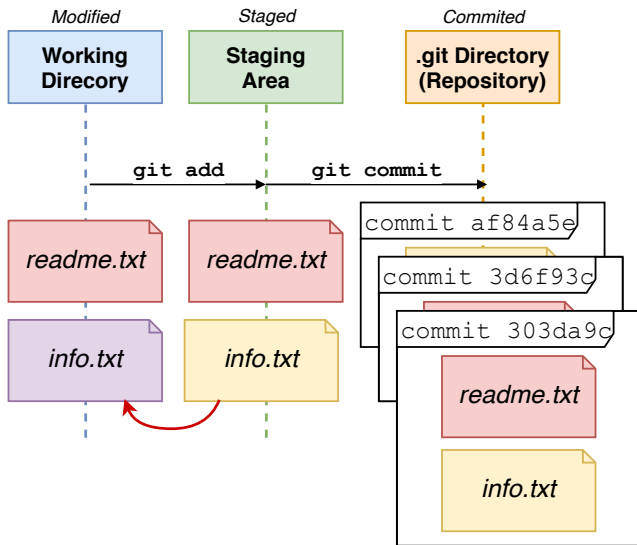
```
to update what will be committed)
```

```
(use "git checkout -- <file>..."
```

```
to discard changes in working directory)
```

```
modified:   info.txt
```

# Locale wijzigingen ongedaan maken





# Locale wijzigingen ongedaan maken

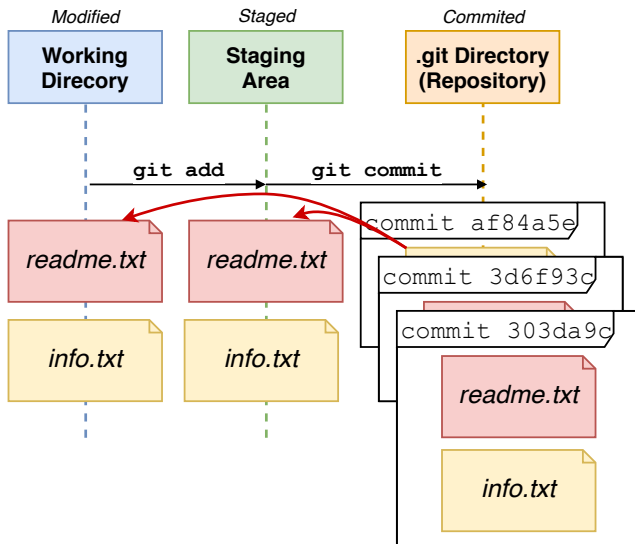
```
$ git checkout -- info.txt
```

```
$ git status
```

```
On branch master
```

```
nothing to commit, working tree clean
```

# Een bestand naar een eerdere revisie zetten



# Een bestand naar een eerdere revisie zetten

```
$ git log --oneline
303da9c (HEAD -> master) info.txt toegevoegd
3d6f93c readme updated
af84a5e readme.txt added

$ git checkout af84a5e -- readme.txt
```

## De situatie na de checkout

```
$ git status
```

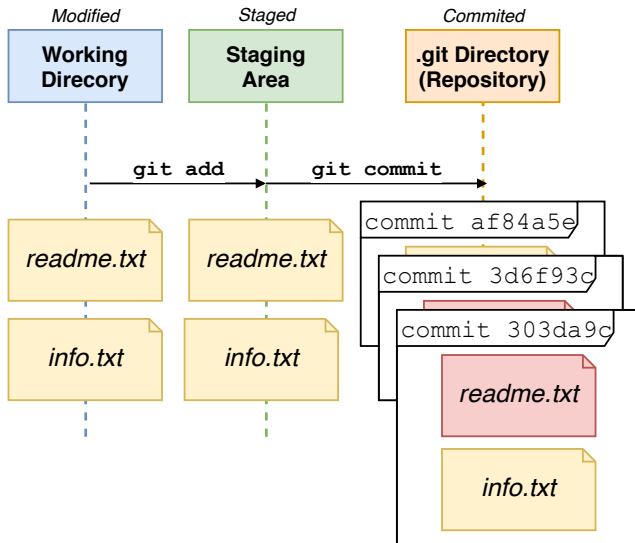
```
On branch master
```

```
Changes to be committed:
```

```
(use "git reset HEAD <file>..." to unstage)
```

```
    modified:   readme.txt
```

# De situatie na de checkout



## De situatie na de checkout: wat nu?

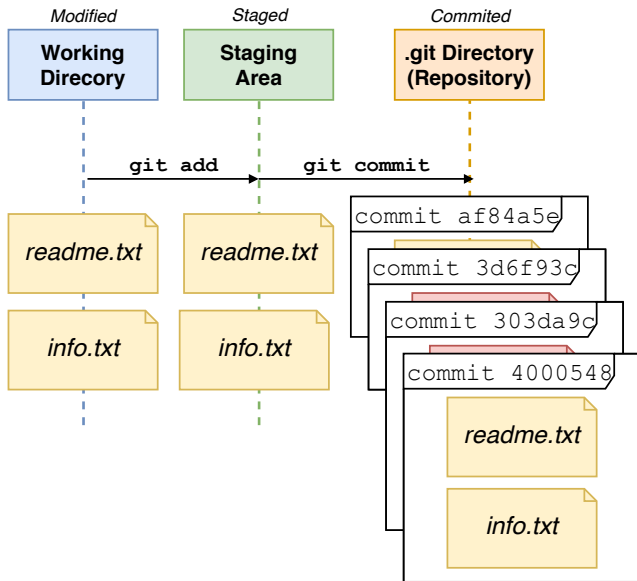
Of: nieuwe wijzigingen aanbrengen,

vervolgd door `git add .` & `git commit`

Of: direct commiten

```
$ git commit -m "readme.txt naar originele revisie"
[master 4000548] readme.txt naar originele revisie
1 file changed, 1 deletion(-)
```

## De situatie na de checkout en commit



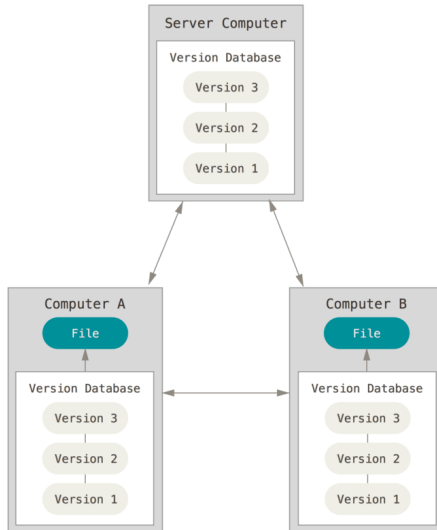
## Even controlleren...

```
$ git log --oneline
4000548 (HEAD -> master) readme.txt naar originele revisie
303da9c info.txt toegevoegt
3d6f93c readme updated
af84a5e readme.txt added
```



## Git remote

# Waarom remote

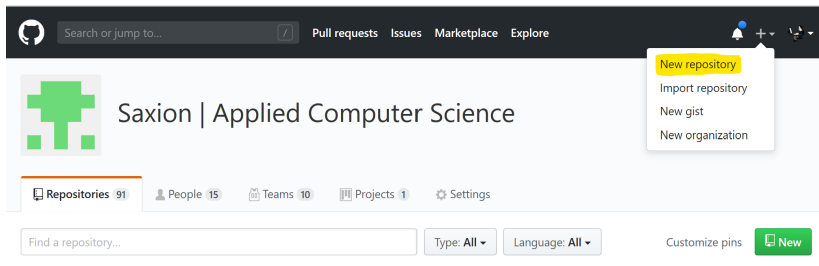


# GitHub

<https://github.com/>

- Meest populaire git-hosting service
- Gratis voor iedereen (met beperkingen)
- Gratis voor onderwijs (zonder beperkingen)
- GitHub Classroom (verder)

# Een nieuwe repository op GitHub aanmaken



# Details invoeren

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner

Repository name \*



SaxionACS ▾

/

git\_workshop



Great repository names are short and memorable. Need inspiration? How about **potential-invention**?

Description (optional)

This is a git workshop using markdown and beamer



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾



# Public vs. Private

## Public

Iedereen kan de bestanden zien en kopiëren (maar niet wijzigen).

Alleen de eigenaar en de teamleden kunnen de bestanden aanpassen.

## Private

Alleen de eigenaar en de teamleden kunnen de bestanden zien, kopiëren of aanpassen.

# Lokaal en remote koppelen

## Quick setup — if you've done this kind of thing before



Set up in Desktop

or

HTTPS

SSH

[https://github.com/SaxionACS/git\\_workshop.git](https://github.com/SaxionACS/git_workshop.git)

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

## ...or create a new repository on the command line

```
echo "# git_workshop" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/SaxionACS/git_workshop.git
git push -u origin master
```



## Een bestande repo met een remote koppelen


```
$ git remote add origin  
    https://github.com/SaxionACS/git_workshop.git  
$ git push -u origin master
```



## Alternatief: remote clonen naar een nieuwe map

```
$ mkdir my_project
$ cd my_project
$ git clone
  https://github.com/SaxionACS/git_workshop.git
```

# Is het gelukt?


[SaxionACS / git\\_workshop](#)

Unwatch 1
Star 0
Fork 0


Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings



This is a git workshop using markdown and beamer



[Manage topics](#)

4 commits
1 branch
0 releases
1 contributor

Branch: master
New pull request
Create new file
Upload files
Find File
Clone or download


**zaldawid**
readme.txt naar originele revisie
Latest commit 4000548 18 hours ago

 info.txt	info.txt toegevoegt	19 hours ago
 readme.txt	readme.txt naar originele revisie	18 hours ago


**readme.txt**


```
readme.
```

# Is het gelukt?

```
$ git remote -v
```

```
origin
```

```
https://github.com/SaxionACS/git_workshop.git (fetch)
```

```
origin
```

```
https://github.com/SaxionACS/git_workshop.git (push)
```

# Wat nou

Het hangt eraf:

- Solo gebruiker
- Team gebruiker

## Remote voor een solo gebruiker

```
$ git pull
[bewerk bestanden]
$ git status
$ git add .
$ git commit -m "... "
$ git push
```

- pull: Wijzigingen remote -> lokaal
- push: Wijzigingen lokaal -> remote

# Remote voor een team: collaborators

SaxionACS / git\_workshop

Unwatch 1Star 0Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

Settings

Options

**Collaborators & teams**

Branches

Webhooks

Notifications

Integrations & services

Deploy keys

Alerts

Moderation

Interaction limits

Teams

Create new team

No teams have been given access to this repository yet. Use the form below to add a team.

Add a team

Collaborators

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

## Remote voor een team: initializatie

Iedereen moet zijn lokale repo naar dezelfde remote verwijzen.

Één person zorgt voor de initialisatie van de remote repository.

Dan:

```
$ git clone  
https://github.com/SaxionACS/git_workshop.git
```

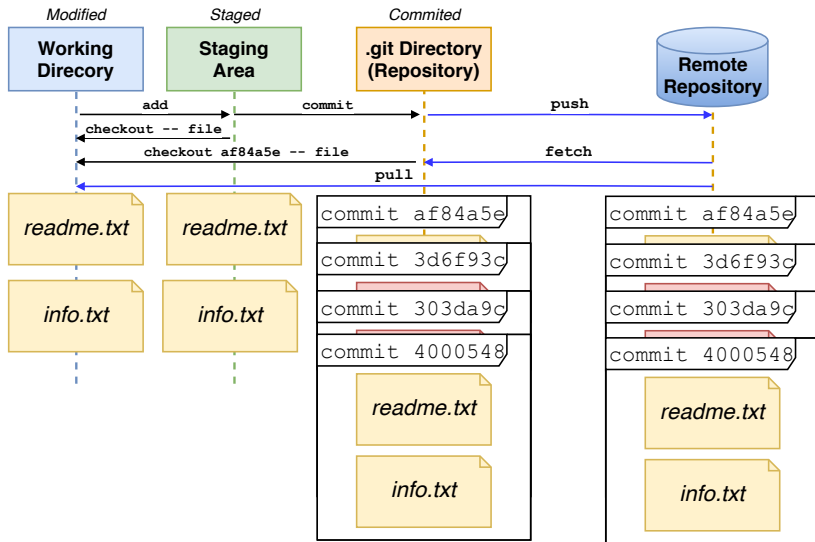
## Remote voor een team: workflow

```
$ git pull --rebase
[bewerk bestanden]
$ git status
$ git add .
$ git commit -m "...".
$ git pull --rebase
$ git push
```

--rebase zorgt voor “eenvoudige” historiek



# Werken met remote

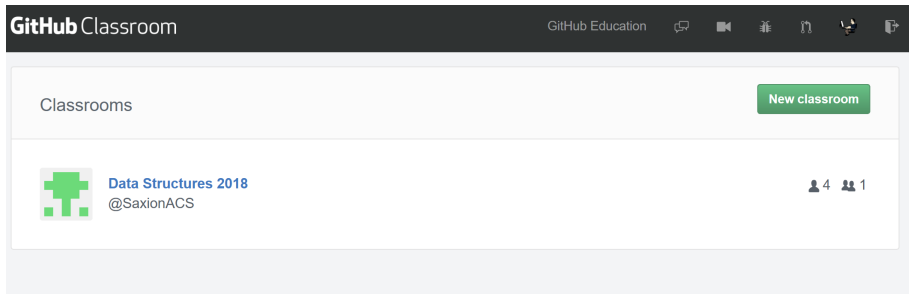


# GitHub voor onderwijs

# GitHub voor docenten

- Koppel aan emailadres van je school aan je GitHub account
- Registreer je op GitHub Education:  
<https://education.github.com/teachers>
- Maak nieuwe Github “organisatie” aan
  - bijv. naam van je school of vak of project
- Vraag korting aan:  
[https://education.github.com/discount\\_requests/new](https://education.github.com/discount_requests/new)
- Je kan ook korting voor jouw persoonlijke account aanvragen.

# GitHub classroom



<https://classroom.github.com>

# Wat kun je ermee

- Opdrachten voor studenten zetten:  
<https://github.com/SaxionACS/AdventOfCode>
- Presentaties maken;): [https://github.com/SaxionACS/git\\_workshop](https://github.com/SaxionACS/git_workshop)

# Mogelijkheden

- Feedback geven
  - issues, pull requests, commentaar
- Verbeteringen aanbrengen
  - online bestanden aanpassen
- Vooruitgang opvolgen
  - commit log (wie, wat, wanneer)
- Samenwerking beoordelen
  - commit log (wie, wat, wanneer)
- Discussies opvolgen
- Project plannen

# Demo & vragen

Tijd voor demo & vragen