



Git

1. Inleiding

1.1 Git in 2 minuten

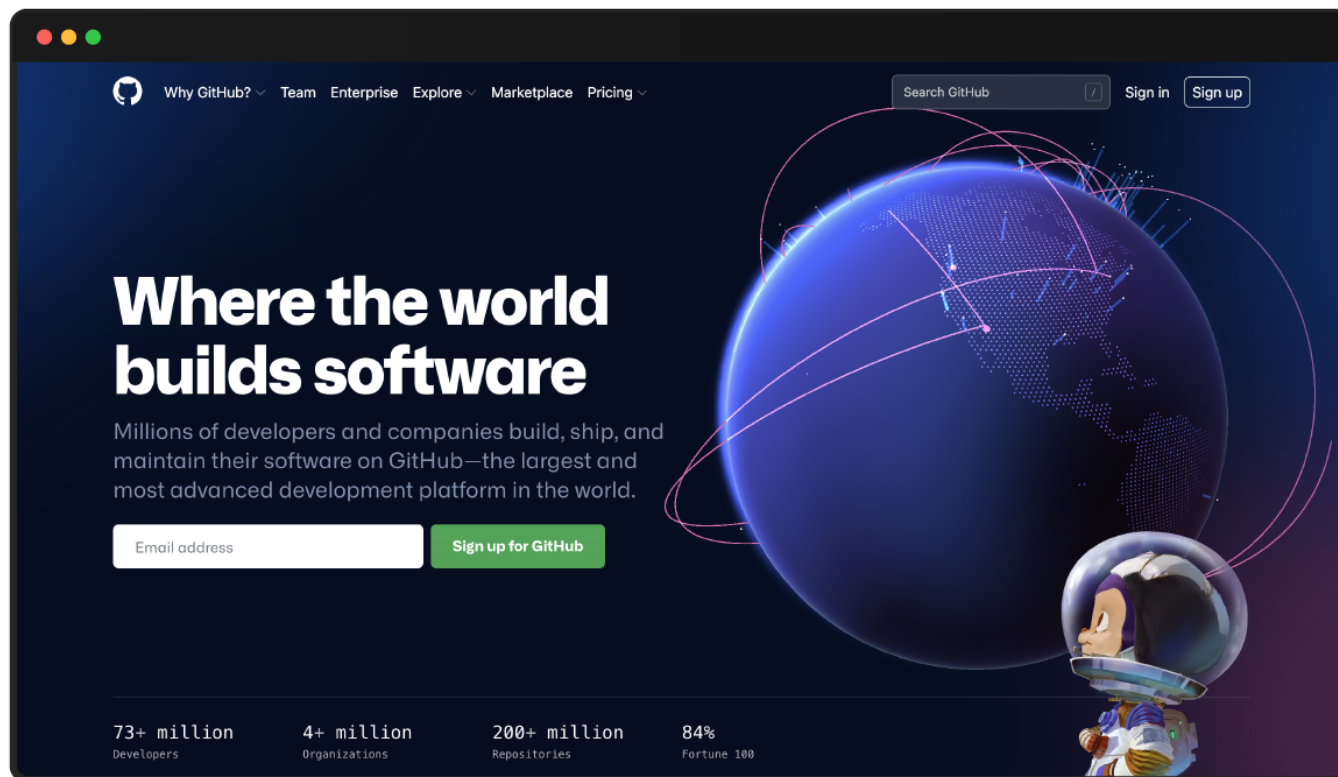
Git is een gratis en open source *version control system* dat ons helpt om versiebeheer in onze projecten te managen. Dit maakt niet alleen onze workflow een stuk gemakkelijker, het zorgt er ook voor dat we met meerdere ontwikkelaars *tegelijkertijd* aan één project kunnen werken. Het is dan ook niet gek dat meer dan 90% van alle software projecten in de wereld worden beheerd met Git.

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

1.2 GitHub account aanmaken

De namen **Git** en **GitHub** lijken wellicht op elkaar, maar deze concepten zijn niet hetzelfde. Git is de onderliggende techniek om versiebeheer mogelijk te maken in onze projecten. GitHub is het platform dat we gebruiken om onze code met behulp van Git online, op een server te plaatsen. Om hier gebruik van te kunnen maken, is het handig om eerst een account aan te maken voor je het installatieproces begint.

1. Ga naar www.github.com en kies voor "Sign Up";
2. Maak een nieuw account aan. Doe dit *niet* met jouw *@novi-education* email-adres, maar met een persoonlijk e-mailadres. De reden hiervoor is dat je niet voor eeuwig bij ons zult studeren, maar je jouw GitHub-account wel je leven lang met je meeneemt!



Gelukt? Dan kun je beginnen aan de installatie!

2. Installatie (Windows)

2.1 Git installeren

In onderstaande video wordt uitgelegd hoe je Git installeert op jouw Windows besturingssysteem en hoe je kunt checken of er al een versie geïnstalleerd is. Alle stappen (en commando's) in de video worden hieronder ook uitgelegd in tekst. Zo kun je ook de gebruikte commando's makkelijk terugvinden!

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

Om het invoeren van commando's makkelijker te maken, open je een leeg project in jouw IDE naar keuze. In de video maken we gebruik van WebStorm. Ben je bekend met het gebruik van de commandline? Dan kun je ook de Command Prompt of Windows Powershell gebruiken.

1. Checken of Git geïnstalleerd moet worden

Controleer of Git al op jouw besturingssysteem door het volgende commando in de terminal in te voeren:

```
git help
```

Druk vervolgens op Enter. Krijg je een rode foutmelding te zien die lijkt op "Git is not recognized as an internal command"? Dit betekent dat het nog niet

geïnstalleerd is. Ga verder bij stap 2: *Installatie*.

Krijg je geen rode foutmelding te zien, maar een opsomming van Git-commando's die je kunt gebruiken? Nice! Dan staat Git al op je besturingssysteem en kun je verder gaan met het instellen van globals in de *volgende* paragraaf.

2. Installatie

Je kunt de Git-installer downloaden via [deze website](#). Kies voor "Windows" en vervolgens voor "64 bit". Wanneer de *installer* klaar is met downloaden, open je het .exe bestand via de map *Downloads*, door erop te dubbelklikken.

Blijf in het installatie-scherm telkens op "Next" klikken. Je hoeft geen instellingen aan te passen. *Let op*: Git zal standaard voorstellen om het op de C-schijf te installeren, breng hier geen veranderingen in aan.

Wanneer de installatie voltooid is, is het belangrijk om je machine opnieuw op te starten. Je kunt hetzelfde resultaat ook bereiken door alle IDE's die je nog open hebt staan af te sluiten en vervolgens ook Windows Verkenner te herstarten via Taak Beheer.

3. Checken of nu Git correct geïnstalleerd is

Na het herstarten open je opnieuw jouw IDE naar keuze. Open de terminal en controleer of Git geïnstalleerd is door het volgende commando in te voeren:

`git help`

Druk vervolgens op Enter. Krijg je een opsomming van Git-commando's die je kunt gebruiken? Dan is het gelukt! Je kunt verder met de configuratie in de volgende paragraaf.

Krijg je een rode foutmelding te zien die lijkt op *"Git is not recognized as an internal command"*? Dan is er iets mis gegaan. Probeer eerst je besturingssysteem opnieuw op te starten. Krijg je na het herstarten en opnieuw invoeren van bovenstaand commando nog steeds een foutmelding te zien? Installeer dan Git volgens bovenstaande stappen opnieuw.

2.2 Globals en SSH-key instellen

In onderstaande video wordt uitgelegd hoe je jouw Git globals (globale variabelen) kunt instellen en hoe je een SSH-key aanmaakt. Dit heb je nodig om jouw GitHub identiteit te koppelen aan jouw machine. Alle stappen (en commando's) in de video worden hieronder ook uitgelegd in tekst. *Let op*: als het installeren van Git nog niet gelukt is, kun je niet verder met de stappen in deze paragraaf. Vraag eerst om hulp bij een docent of SME'er.

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

1. Globals instellen

Login op www.Github.com en check welk e-mailadres en gebruikersnaam je voor dit account hebt gekozen. Stel dat onze GitHub gebruikersnaam pietpieters is en ons GitHub e-mailadres piet_pieters@gmail.com. Dan voeren we de volgende twee commando's in, in de terminal:

```
git config --global user.name "pietpieters"
```

Druk nu op Enter. Er zal daarna niets gebeuren, je krijgt simpelweg een nieuwe regel te zien waarop je het volgende commando invoert:

```
git config --global user.email "piet_pieters@gmail.com"
```

Druk opnieuw op Enter. We krijgen wederom geen duidelijke bevestiging dat dit *gelukt*, of *mislukt* is. Dit controleren we daarom met het volgende commando:

```
git config --list
```

Je krijgt nu een lijst te zien. Blijf telkens opnieuw op Enter drukken, tot je aan het eind van de lijst bent en geen nieuwe variabelen meer te zien krijgt. Wees je ervan bewust dat je nu in een andere terminal-weergave zit. Als je zometeen weer commando's wil gaan invoeren, zul je uit deze weergave moeten stappen door het commando `:q` in te voeren en op Enter te drukken. Die `q` staat voor *Quit*. Als het goed is, heb je dit voorbij zien komen:

```
user.name=pietpieters
user.email=piet_pieters@gmail.com
```

Stonden jouw gebruikersnaam en email-adres in de lijst? Dan is het gelukt! Je kunt nu verder met *stap 2: SSH-key aanmaken*.

Zie je géén gebruikersnaam of e-mailadres, of slechts één van de twee? Herhaal alle commando's in deze paragraaf dan, net zo lang tot de juiste informatie in de lijst staat.

2. SSH-key aanmaken en koppelen

Nu kunnen we een SSH-key aanmaken. Een SSH-key bestaat altijd uit twee delen: een *public key* en een *private key*. De private key staat op ons besturingssysteem en delen we met niemand. De public-key geven we door aan GitHub, zodat we niet bij elke data-uitwisseling hoeven te bewijzen wie we zijn (door gebruikersnamen en wachtwoorden in te voeren).

Open jouw IDE naar keuze. Open de terminal en maak een nieuwe SSH-key aan door het volgende commando in te voeren:

```
ssh-keygen -t ed25519
```

Druk vervolgens op Enter. Kijk goed naar de melding in de terminal. Hier hoort nu: "Generating public/private ed25519 keypair" te staan. Indien dit niet zo is, heb je een spelfout gemaakt en zul je het opnieuw moeten proberen.

1. De terminal zal je nu de volgende vraag stellen: "Enter file in which to save the key:". De standaard locatie die wordt voorgesteld, is in jouw gebruikers-map op de C-schijf. Hier gaan we mee akkoord, dus drukken we op Enter.
2. Vervolgens wordt je gevraagd om een *passphrase* (een soort wachtwoord) te bedenken. Dit biedt uiteraard meer veiligheid, maar zorgt er ook voor dat je iedere keer als je iets met Git doet, om je wachtwoord wordt gevraagd. Voor de opleiding is dit niet zo handig, **dus voeren we geen passphrase in**. Je hoeft alleen op Enter te drukken.
3. Ten slotte wordt je gevraagd de (lege) passphrase te bevestigen: "Enter same passphrase again: ". Dus vullen we niets in en drukken we nogmaals op Enter. Je krijgt nu een soort kunstwerkje te zien:

```

Your public key has been saved in C:\Users\v.vanholten/.ssh/id_ed25519.pub.
The key fingerprint is:
SHA256:oSTyqMJqkS/x7UFJeAZEEEq5sQTCr4Bwc5MnXNEGsoM novi\v.vanholten@NOVI-UT0060
The key's randomart image is:
+--[ED25519 256]--+
| =+*=.0++      |
| =o.0o. o      |
| =.E++B...     |
| oo =++... .   |
| . + .o. S     |
| .* .          |
| o.= ..        |
| oo o ..       |
| o . ..        |
+-----[SHA256]-----+
PS C:\Users\v.vanholten\WebstormProjects\git-installatie>

```

1. Open nu *Windows verkenner* > *C-schijf* > *Gebruikers* en vervolgens de map met de gebruikersnaam van jouw besturingssysteem. Hier vind je een map genaamd `.ssh`. Open deze map.
2. Je zult hier twee bestanden in vinden, waarvan je het `id_ed25519.pub` bestand wil openen met Kladblok. Dit doe je door met de rechtermuisknop op het bestand te klikken > *Openen met...* > *Meer Apps* > *Kladblok*.
3. Kopieer de volledige inhoud van dit bestand.
4. Log in op [www.GitHub.com](https://www.github.com) en ga vervolgens naar de instellingen (*Settings* > *SSH & GPG keys*). Druk dan op de grote groene knop met "New SSH key" en plak jouw public key in het grote veld. Geef jouw key ook een naam die beschrijft welk apparaat deze key gebruikt, zoals "Pieters Asus laptop" of "Pieters Desktop".
5. Druk op de knop "Add SSH key". Vervolgens zul je de SSH key tussen jouw keys zien staan. Dan is het gelukt!

3. Installatie (Mac)

3.1 Git installeren

In onderstaande video wordt uitgelegd hoe je Git installeert op jouw Mac besturingssysteem en hoe je kunt checken of er al een versie geïnstalleerd is. Alle stappen (en commando's) in de video worden hieronder ook uitgelegd in tekst. Zo kun je ook de gebruikte commando's makkelijk terugvinden!

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

Om het invoeren van commando's makkelijker te maken, open je een leeg project in jouw IDE naar keuze. In de video maken we gebruik van WebStorm. Ben je bekend met het gebruik van de commandline? Dan kun je ook de terminal of iTerm2 gebruiken.

1. Checken of Git geïnstalleerd moet worden

Controleer of Git al op jouw besturingssysteem door het volgende commando in de terminal in te voeren:

```
git --version
```

Druk vervolgens op Enter. Je kunt nu één van drie dingen te zien krijgen:

1. Een rode melding met: "Git is not recognized as an internal command". Dit betekent dat het nog niet geïnstalleerd is. Ga verder bij stap 2: *Installatie*.
2. Een versienummer, zoals `git version 2.33.1` of `2.13.7 (apple-git)`. Is deze versie hoger dan 2.17.0? Nice! Dan staat de juiste Git-versie al op je besturingssysteem en kun je verder gaan met het instellen van globals in de *volgende* paragraaf. Is het versienummer lager dan 2.17.0? Ga verder bij stap 2: *Installatie*.
3. Een melding of pop-up waarin je wordt gevraagd om de developer tools Xcode te installeren. Je kunt dit oplossen door deze applicatie te downloaden en installeren via de App Store, maar Xcode is best een grote applicatie die je niet nodig hebt zolang je geen native MacOS of iOS apps ontwikkelt. Het scheelt je wat geheugen als je in plaats daarvan handmatig een nieuwe versie van Git installeert. Ga verder bij stap 2: *Installatie*.

2. Installatie

Je kunt de Git-installer downloaden via [deze website](#). Kies voor "Mac" en vervolgens bij het kopje "Binary Installer" voor de link "The latest version is 2.33.0". De installer zal binnen vijf seconden worden gedownload. Als deze klaar is, open je het `.dmg` bestand via de map *Downloads*, door erop te dubbelklikken.

Het is mogelijk dat je de melding "Deze app is afkomstig van een onbekende ontwikkelaar en kan niet worden geopend" te zien krijgt. Dit kun je oplossen door jouw Systeemvoorkeuren te openen > *Beveiliging en Privacy* > *Algemeen* > "Open toch".

Vervolgens krijg je het installatiescherm te zien. Blijf hierin telkens op "Next" klikken. Je hoeft geen instellingen aan te passen. *Let op*: Git zal standaard voorstellen om het op de C-schijf te installeren, breng hier geen veranderingen in aan.

Wanneer de installatie voltooid is, is het belangrijk om je machine opnieuw op te starten.

3. Checken of nu Git correct geïnstalleerd is

Na het herstarten open je opnieuw jouw IDE naar keuze. Open de terminal en controleer of Git geïnstalleerd is door het volgende commando in te voeren:

```
git --version
```

Druk vervolgens op Enter. Er kunnen nu drie dingen gebeuren:

1. Je krijgt een versienummer te zien die hoger is dan 2.17.0. Dan is het gelukt! Je kunt verder met de configuratie in de volgende paragraaf.
2. Je krijgt opnieuw de foutmelding "Git is not recognized as an internal command" te zien. Dan is er iets mis gegaan. Installeer dan Git volgens bovenstaande stappen opnieuw.
3. Je wordt *alsnog* gevraagd om Xcode te installeren. Dan is jouw besturingssysteem op dit moment een beetje in de war. Er is nu namelijk een apple-git en een gewone git-versie geïnstalleerd, dus we moeten nog even laten weten dat we geen gebruik meer willen maken van de apple-git versie (die Xcode nodig heeft). We kunnen dit oplossen door de volgende commando's in de terminal in te voeren:

```
echo "PATH=/usr/local/git/bin:$PATH" >> ~/.bash_profile
```

```
source ~/.bash_profile
```

3.2 Globals en SSH-key instellen

In onderstaande video wordt uitgelegd hoe je jouw Git globals (globale variabelen) kunt instellen en hoe je een SSH-key aanmaakt. Dit heb je nodig om jouw GitHub identiteit te koppelen aan jouw machine. Alle stappen (en commando's) in de video worden hieronder ook uitgelegd in tekst. **Let op:** als het installeren van Git nog niet gelukt is, kun je niet verder met de stappen in deze paragraaf. Vraag eerst om hulp bij een docent of SME'er.

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

1. Globals instellen

Log in op www.github.com en check welk e-mailadres en gebruikersnaam je voor dit account hebt gekozen. Stel dat onze GitHub gebruikersnaam pietpieters is en ons GitHub e-mailadres piet_pieters@gmail.com. Dan voeren we de volgende twee commando's in, in de terminal:

```
git config --global user.name "pietpieters"
```

Druk nu op Enter. Er zal daarna niets gebeuren, je krijgt simpelweg een nieuwe regel te zien waarop je het volgende commando invoert:

```
git config --global user.email "piet_pieters@gmail.com"
```

Druk opnieuw op Enter. We krijgen wederom geen duidelijke bevestiging dat dit *gelukt*, of *mislukt* is. Dit controleren we daarom met het volgende commando:

```
git config --list
```

Je krijgt nu een lijst te zien. Blijf telkens opnieuw op Enter drukken, tot je aan het eind van de lijst bent en geen nieuwe variabelen meer te zien krijgt. Wees je ervan bewust dat je nu in een andere terminal-weergave zit. Als je zometeen weer commando's wil gaan invoeren, zul je uit deze weergave moeten stappen door het commando :q in te voeren en op Enter te drukken. Die q staat voor *Quit*. Als het goed is, heb je dit voorbij zien komen:

```
user.name=pietpieters
user.email=piet_pieters@gmail.com
```

Stonden jouw gebruikersnaam en email-adres in de lijst? Dan is het gelukt! Je kunt nu verder met *stap 2: SSH-key aanmaken*.

Zie je géén gebruikersnaam of e-mailadres, of slechts één van de twee? Herhaal alle commando's in deze paragraaf dan, net zo lang tot de juiste informatie in de lijst staat.

2. SSH-key aanmaken en koppelen

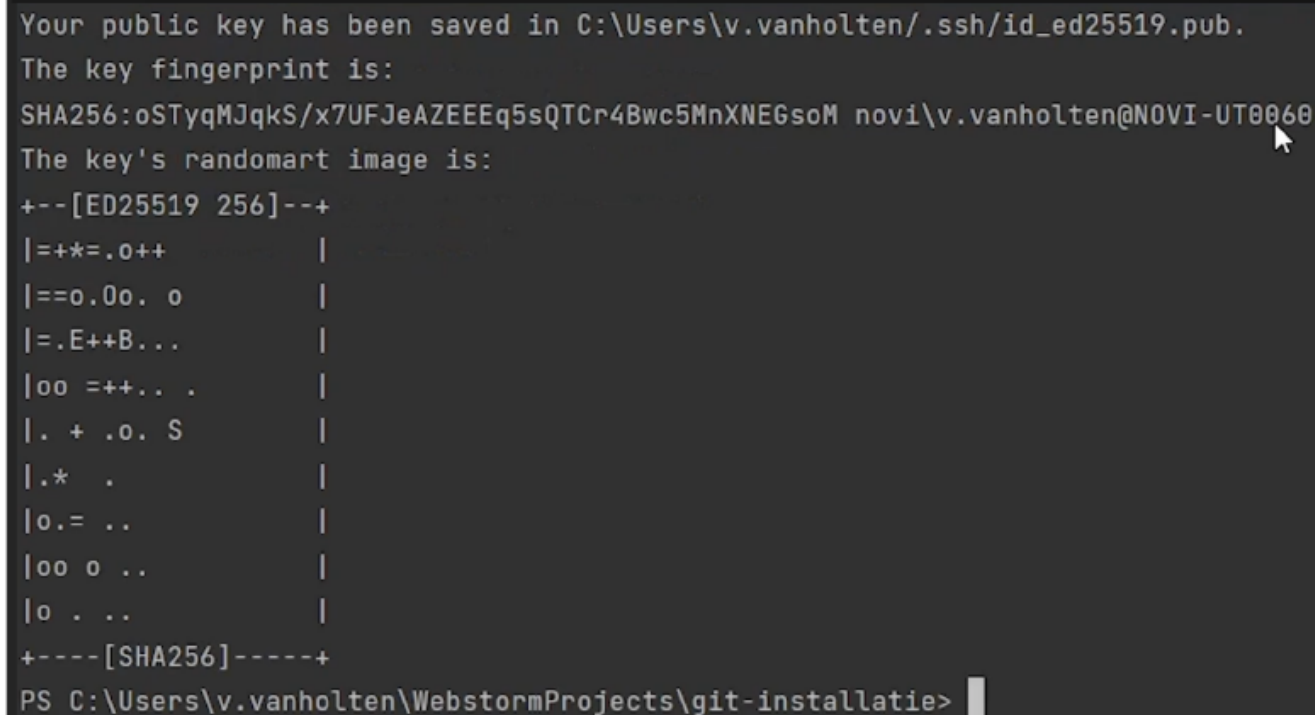
Nu kunnen we een SSH-key aanmaken. Een SSH-key bestaat altijd uit twee delen: een *public key* en een *private key*. De private key staat op ons besturingssysteem en delen we met niemand. De public-key geven we door aan GitHub, zodat we niet bij elke data-uitwisseling hoeven te bewijzen wie we zijn (door gebruikersnamen en wachtwoorden in te voeren).

Open jouw IDE naar keuze. Open de terminal en maak een nieuwe SSH-key aan door het volgende commando in te voeren:

```
ssh-keygen -t ed25519
```

Druk vervolgens op Enter. Kijk goed naar de melding in de terminal. Hier hoort nu: "Generating public/private ed25519 keypair" te staan. Indien dit niet zo is, heb je een spelfout gemaakt en zul je het opnieuw moeten proberen.

1. De terminal zal je nu de volgende vraag stellen: "Enter file in which to save the key:". De standaard locatie die wordt voorgesteld, is in jouw gebruikers-map op de C-schijf. Hier gaan we mee akkoord, dus drukken we op Enter.
2. Vervolgens wordt je gevraagd om een *passphrase* (een soort wachtwoord) te bedenken. Dit biedt uiteraard meer veiligheid, maar zorgt er ook voor dat je iedere keer als je iets met Git doet, om je wachtwoord wordt gevraagd. Voor de opleiding is dit niet zo handig, **dus voeren we geen passphrase in**. Je hoeft alleen op Enter te drukken.
3. Ten slotte wordt je gevraagd de (lege) passphrase te bevestigen: "Enter same passphrase again: ". Dus vullen we niets in en drukken we nogmaals op Enter. Je krijgt nu een soort kunstwerkje te zien:



```
Your public key has been saved in C:\Users\v.vanholten/.ssh/id_ed25519.pub.  
The key fingerprint is:  
SHA256:oSTYqMJqkS/x7UFJeAZEEEq5sQTCr4Bwc5MnXNEGsoM novi\v.vanholten@NOVI-UT0060  
The key's randomart image is:  
+--[ED25519 256]--+  
| =+* = .o++      |  
| =o.o.o. o       |  
| =.E++B...      |  
| oo =++.. .      |  
| . + .o. S       |  
| .* .           |  
| o.= ..         |  
| oo o ..        |  
| o . ..         |  
+----[SHA256]-----+  
PS C:\Users\v.vanholten\WebstormProjects\git-installatie>
```

1. We kopiëren de public key door het volgende commando in te voeren:

```
pbcopy < ~/.ssh/id_ed25519.pub
```


1. Log in op www.github.com en ga vervolgens naar de instellingen (*Settings > SSH & GPG keys*). Druk dan op de grote groene knop met "New SSH key" en plak jouw public key in het grote veld. Geef jouw key ook een naam die beschrijft welk apparaat deze key gebruikt, zoals "Pieters iMac" of "Pieters MacBook Pro (M1)".
2. Druk op de knop "Add SSH key". Vervolgens zul je de SSH key tussen jouw keys zien staan. Dan is het gelukt!

4. Git in gebruik

4.1 Comitten en pushen

In onderstaande video wordt uitgelegd hoe git precies werkt, welke commando's je nodig hebt om basale acties uit te voeren en wat ze precies betekenen. Alle gebruikte commando's in de video kun je onderaan deze paragraaf nog eens nalezen. *Let op:* deze video is gericht op het *uitleggen* van de concepten. De praktische toepassing van alle commando's wordt behandeld in de volgende paragrafen.

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

Basis commando's

Wanneer je wil weten of jouw bestanden wel of niet *gestaged* zijn, gebruik je:

```
git status
```

Je kunt alle gewijzigde bestanden in één keer stagen met het commando:

```
git add .
```

... of je *staged* alleen specifieke bestanden, door de punt te vervangen door de letterlijke bestandsnaam:

```
git add bestandsnaam.css
```

Wanneer je tevreden bent over de bestanden die je *gestaged* hebt, kun je ze verpakken in een commit. Dit doe je met het volgende commando, waarbij je altijd een berichtje tussen de twee aanhalingstekens schrijft:

```
git commit -m "Loginfunctionaliteit aangepast en huisstijl kleuren toegevoegd"
```

Ten slotte is het mogelijk om alle openstaande commits naar de repository op GitHub te pushen. *Let op:* hiervoor moet er wel een repository gekoppeld zijn. Hoe je dit doet, leer je in de volgende paragrafen.

```
git push origin main
```

4.2 Praktisch: nieuw project opzetten met git

In onderstaande video wordt gedemonstreerd hoe je een nieuw (frontend) project opzet met Git. Alle gebruikte commando's in de video kun je onderaan deze paragraaf nog eens nalezen.

Zie je geen video verschijnen? Klik [hier](#) om de video in een nieuw tabblad te openen.

Initialisatie en opzet

Je initialiseert git in een project door middel van het commando:

```
git init
```

Vanaf dat moment wordt er versiebeheer bijgehouden in jouw projectmap en ben je in staat om commits te maken. Daarna is het ook belangrijk om een `.gitignore` bestand aan te maken in het project (*Rechtermuisknop op de projectmap > New > File*). In dit bestand houd je een lijst bij van bestanden en mappen die wel in jouw project-map staan, maar niet mogen worden meegepusht naar GitHub. Wanneer je een JetBrains editor gebruikt, wordt er bijvoorbeeld een verborgen `.idea`-map aangemaakt waar jouw editor-settings in staan. Dat is natuurlijk totaal niet interessant voor collega-developers. Daarom voegen we deze onder andere toe aan het `.gitignore` bestand:

```
/.idea
```

In de toekomst wil je hier nog meer aan toevoegen, zoals bijvoorbeeld `node_modules` in JavaScript- en React-projecten en de `target`- en `out`-map in Java- en SpringBoot-projecten.

Commits maken

Daarna kun je beginnen met het aanmaken van een aantal bestanden. In een frontend project zal dat bijvoorbeeld een `index.html` en een `styles.css` zijn, waarin je vervolgens een aantal aanpassingen maakt. Benieuwd hoe het staat met je bestanden? Gebruik dan het commando:

```
git status
```

Vervolgens kun je al jouw bestanden *stagen* en committen door:

```
git add .
```

```
git commit -m "Leuke opzet gemaakt met a en b en c"
```

Als dit gelukt is, krijg je een soortgelijke melding: "[main 2501693] Leuke opzet gemaakt met a en b en c". *Let op:* wanneer je geen bestanden hebt *gestaged* kun je ook geen commit maken. Niemand heeft immers iets aan een lege verhuisdoos.... Je krijgt dan ook melding in de terminal die je wijst op de *ongestagede* bestanden. Let dus altijd goed op de feedback die je terugkrijgt in de terminal.

Ben je benieuwd hoeveel commit's je al hebt gemaakt in dit project? Dan kun je het volgende commando gebruiken:

```
git log
```

Repository aanmaken, koppelen en pushen

Een repository maak je aan via [www.GitHub.com](https://www.github.com). Daarna krijg je een overzicht van instructies om de repository te koppelen aan een lokaal project. Zorg altijd dat het knopje SSH is aangeklikt bovenaan de pagina. Kies hierbij altijd voor de instructies bij het kopje "...or push an existing repository from the command line".

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

or

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 "# test" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:hogeschoolnovi/test.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:hogeschoolnovi/test.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Voer de drie voorgestelde commando's één voor één in. Na het laatste commando worden al jouw lokale commits naar de repository gepusht. Wanneer je de webpagina ververs, zul je daar al jouw bestanden zien staan!

Hierna kun je naar hartelust aanpassingen maken, commits doen en die vervolgens weer pushen met het commando:

```
git push origin main
```