Git (GitHub) 1

Les 1 | Wat is Git?

? Leerdoelen

- Je kunt uitleggen wat Git is.
- Je kent het verschil tussen Git en GitHub.
- Je begrijpt de belangrijkste Git-begrippen.
- Je weet waarom we in deze module vooral met Source Control in VS Code werken.

? Uitleg

Stel: je werkt aan een website en maakt een fout. Je hebt overal overheen opgeslagen en kunt niet meer terug. Git lost dit op. Git is een **versiebeheersysteem**: elke keer dat je iets nieuws hebt gemaakt voor je applicatie of website, sla je dit in git op (commit). Je hebt nu een **backup** van je applicatie.

Let op: Zie Git als een slimme versiegeschiedenis van je hele project. Je kunt dus terug naar eerdere werkende versies.

Simpel gezegd: Git is Ctrl+Z op steroïden, maar dan voor je hele project.

Jouw project met alle opgeslagen versies sla je met git op, op je PC. Je kunt deze bestanden en versies ook in de cloud opslaan. Vaak doe je dit bij de online dienst GitHub. Dit is handig om backups van alle versies te bewaren, maar ook handig om met andere samen te werken aan één project.

? Git vs. GitHub

afbeelding.png

Overzicht van lokaal werken met Git en online samenwerken via GitHub

Vraag	Git	GitHub
Wat is het?	Software op je laptop	Een website waar je repositories online opslaat
Waar gebruik je het?	Lokaal op je computer	Online in de cloud
Waarvoor?	Versiebeheer	Backup, samenwerken en code delen

? Belangrijke begrippen

Begrip	Betekenis
Repository (repo)	Een projectmap die Git bijhoudt. Eigenlijk je hele project.
Commit	Het opslaan van een versie van je hele project.
Staging area	De wachtrij met bestanden die klaarstaan voor een commit.
Branch	Een aparte tak waarin je veilig kunt experimenteren.
Remote	De online versie van je repo, bijvoorbeeld op GitHub.
Push	Je lokale commits uploaden naar GitHub.
Pull	Wijzigingen ophalen vanaf GitHub.
Clone	Een bestaande online repo downloaden naar je laptop. -> Je haalt dan in één keer een hele applicatie van GitHub naar jouw computer.

? Terminal vs. Source Control

Git kun je gebruiken via de terminal of via het **Source Control**-paneel in VS Code. In deze module gebruiken we vooral Source Control, omdat je daar visueel ziet welke bestanden nieuw, aangepast of verwijderd zijn.

```
git init                 # Nieuwe repo aanmaken
git add .                # Bestanden klaarzetten
git commit -m "..."      # Snapshot opslaan
git push                 # Uploaden naar GitHub
git pull                 # Downloaden van GitHub
git clone <url>          # Repo kopiëren van GitHub
```

Goed om te weten: Je hoeft deze commando’s niet uit je hoofd te leren, maar het is wel handig als je ze herkent.

?? Opdracht – Begrippen begrijpen

Beantwoord de vragen in je eigen woorden. Gebruik volledige zinnen en geef bij minstens één begrip een praktisch voorbeeld uit een project dat je kent.

? Inleveren

1. Een korte uitleg van wat Git is.
2. Een korte uitleg van het verschil tussen Git en GitHub.
3. Een lijst met minimaal 5 begrippen uit deze les met betekenis in je eigen woorden.
4. Een screenshot of notitie waarin je uitlegt waarom Source Control in VS Code handig is.

Les 2 | Git installeren & instellen

? Leerdoelen

- Je kunt Git installeren op Windows.
- Je kunt controleren of Git goed is geïnstalleerd.
- Je kunt je naam en e-mailadres instellen in Git.
- Je kunt een GitHub-account gebruiken en Source Control in VS Code openen.

? Uitleg

Voordat je met versiebeheer kunt werken, moet Git op je computer staan. Daarna vertel je Git wie jij bent. Zo worden je commits gekoppeld aan jouw naam en e-mailadres.

? Stap 1 – Git installeren

1. Ga naar `https://git-scm.com/downloads/win`.
2. Download Git voor Windows.
3. Installeer met de standaardinstellingen.
4. Open de terminal in VS Code met `Ctrl + ``.
5. Controleer de installatie met:

```
git --version
```

Je ziet dan iets als `git version 2.47.1.windows.1`. Dat betekent dat Git werkt.

Als je dit niet ziet, kan het zijn dat je je computer opnieuw moet opstarten! Doe dat en probeer het opnieuw.

? Stap 2 – Git configureren

Git moet weten wie jij bent. Voer in de terminal uit:

```
git config --global user.name "Jouw Naam"
git config --global user.email "jouw.email@voorbeeld.nl"
```

Controleer daarna of alles goed staat:

```
git config --global user.name
git config --global user.email
```

Goed om te weten: Dit hoef je meestal maar één keer te doen op je laptop.

? Stap 3 – GitHub-account

Maak een gratis account aan op `https://github.com/signup` als je die nog niet hebt.

Let op: Kies een professionele gebruikersnaam. Dit is jouw developer-profiel en die laat je later ook aan stagebedrijven of werkgevers zien.

? Stap 4 – Source Control in VS Code

Open Source Control. Source Control (de plek in VS-Code waar je werkt met Git) open je door in je linker balk op onderstaande icoon te klikken. Of door SHIFT - CTRL - G

afbeelding.png

> afbeelding.png
>
> Het Source Control-paneel in VS Code

- **Message**: hier typ je de beschrijving van je commit. (bijvoorbeeld: De achtergrond van de homepage is rood gemaakt.)

- **Commit**: hiermee sla je een snapshot op.
- **Staged Changes**: bestanden die klaarstaan voor commit.
- **Changes**: bestanden die gewijzigd zijn maar nog niet klaarstaan.
- **M, U, D**: modified, untracked en deleted.
- **+** en **−**: bestanden stagen of juist weer uit staging halen.

?? Opdracht – Git klaarzetten

Voer alle stappen uit zodat Git en GitHub klaar zijn voor de volgende lessen.

? Inleveren

1. Screenshot van `git --version` in de VS Code-terminal.
2. Screenshot van de resultaten van `git config --global user.name` en `git config --global user.email`.
3. Screenshot van je GitHub-profielpagina.
4. Screenshot van het Source Control-paneel in VS Code.

Les 3 | Je eerste repository

? Leerdoelen

- Je kunt een nieuwe repository aanmaken.
- Je kunt bestanden stagen en committen via Source Control.
- Je kunt gewijzigde bestanden bekijken in Diff View.
- Je kunt commit-geschiedenis terugvinden.

? Uitleg

In deze les maak je je eerste repository aan. Daarna doorloop je voor het eerst de standaard workflow van Git: **wijzigen → stagen → committen**.

? Stap 1 – Repository aanmaken

Open VS-Code en open een nieuwe folder voor jouw project. Noem deze folder <jouw-naam>-git1.

Source Control (de plek in VS-Code waar je werkt met Git) open je door in je linker balk op onderstaande icoon te klikken. Of door SHIFT - CTRL - G

afbeelding.png

Ga daarna naar Source Control en klik op **Initialize Repository**. Git maakt nu een verborgen `.git` -map aan met alle versie-informatie van je project.

afbeelding.png

Met de iconen in de linkerbalk wissel je tussen 'Explorer' met je bestanden en 'Source Control' voor Git.

afbeelding.png	Explorer voor je bestanden en mappen
afbeelding.png	Source Control voor het werken met Git

? Stap 2 – Eerste commit maken

Maak een bestand `readme.txt` met een klein stukje tekst. Na opslaan zie je het bestand bij **Changes** met de letter **U** van untracked.

afbeelding.png

De drie stappen van een commit in VS Code

1. **Stage:** klik op het **+**-icoon naast het bestand.
2. **Message:** typ bovenaan bijvoorbeeld `Eerste commit`.
3. **Commit:** klik op **Commit** of druk op `Ctrl + Enter`.

Let op: Een goede commit message beschrijft wat je hebt gedaan, bijvoorbeeld *Login pagina toegevoegd* en niet *update*.

? Stap 3 – Workflow begrijpen

afbeelding.png

Van Working Directory naar Staging Area en daarna naar de commit

Je werkt eerst in je projectmap. Daarna kies je welke wijzigingen mee moeten in de commit. Pas als je commit, maakt Git een nieuwe snapshot.

? Stap 4 – Diff View gebruiken

afbeelding.png

Diff View laat precies zien wat is toegevoegd en verwijderd

Wijzig een bestand en klik daarna in Source Control op dat bestand. Links zie je de oude versie, rechts de nieuwe. Rode regels zijn verwijderd, groene regels zijn toegevoegd.

? Stap 5 – Geschiedenis bekijken

afbeelding.png

Commit-geschiedenis in VS Code of via Git Graph

Gebruik de **Timeline** in de Explorer of installeer de extensie **Git Graph** voor een visueel overzicht van je commits.

?? Opdracht – Eerste repo oefenen

Doorloop de volledige workflow en maak minimaal twee commits.

? Inleveren

1. Start een nieuw project in VS-Code door een map aan te maken met de naam `<jouw-naam>-git-oefening` en deze te selecteren (open folder).
2. Maak hier een repo van via **Initialize Repository**.
3. Maak drie bestanden: `index.html`, `style.css` en `script.js` met minimaal één regel code per bestand.
4. Stage alles en commit met de beschrijving `Eerste bestanden`.
5. Wijzig `index.html` door de basis HTML code toe te voegen en wijzig sylte.css, zodat de achtergrond rood wordt. Bekijk de Diff View en maak een tweede commit.

6. Screenshot van het Source Control-paneel met je gestagede bestanden of commit-overzicht.
7. Screenshot van de commit-geschiedenis met minimaal twee commits.

Les 4 | Remote: pushen naar GitHub

? Leerdoelen

- Je kunt een lokale repository publiceren naar GitHub.
- Je begrijpt wat *push*, *pull* en *sync* betekenen.
- Je kunt een bestaande repository clonen.

? Uitleg

Tot nu toe staat je code alleen op je eigen computer. In deze les koppel je je project aan GitHub, zodat je online backup hebt en makkelijk kunt samenwerken.

? GitHub koppelen aan VS Code

1. Open je project in VS Code.
2. Ga naar Source Control.
3. Klik op **Publish Branch**.
4. Kies **Publish to GitHub Public Repository**.
5. Log in met je GitHub-account als VS Code daarom vraagt.

afbeelding.png

Publish Branch en Sync Changes in VS Code

VS Code maakt daarna automatisch een repository op GitHub en pusht je code online.

? Sync Changes begrijpen

Na de eerste publicatie verandert de knop meestal in **Sync Changes**. Die knop doet twee dingen:

- **Push (↑)**: lokale commits uploaden naar GitHub.
- **Pull (↓)**: wijzigingen ophalen vanaf GitHub.

Dagelijkse workflow: maak een commit en klik daarna op **Sync Changes**.

? Klonen van een repo

Je kunt ook een bestaand project downloaden.

1. Open VS Code zonder project.
2. Klik op **Clone Repository**.
3. Plak de GitHub-URL en kies een map op je laptop.

Via de terminal kan dat ook:

```
git clone https://github.com/NAAM/REPO.git
```

?? Opdracht – Online zetten

Publiceer je oefenproject naar GitHub en controleer of je wijziging online zichtbaar wordt.

? Inleveren

1. Publiceer je project via **Publish Branch**.
2. Screenshot van je repository op GitHub met je bestanden.
3. Maak lokaal een wijziging, commit die en klik op **Sync Changes**.
4. Screenshot van GitHub waarop de nieuwe wijziging zichtbaar is.

Les 5 | Branches

? Leerdoelen

- Je begrijpt wat een branch is.
- Je kunt een nieuwe branch maken in VS Code.
- Je kunt wisselen tussen branches.
- Je kunt een branch samenvoegen met `main`.

? Uitleg

Een branch is een aparte versie van je project. Je kunt hier veilig in werken zonder dat je meteen de hoofdversie aanpast. Die hoofdversie heet meestal `main`.

Branches zijn handig als je iets nieuws wilt maken of testen. Bijvoorbeeld een nieuwe pagina, een nieuw menu of een andere layout. Als het goed werkt, kun je de branch later weer samenvoegen met `main`. Dit samenvoegen noemen we **mergen**.

afbeelding.png

Branches maken experimenteren mogelijk zonder risico voor de hoofdbranch

? Nieuwe branch maken

afbeelding.png

Het branch-menu linksonder in VS Code

1. Klik linksonder in VS Code op de huidige branchnaam. Vaak is dit `main`.
2. Kies **Create new branch...**.
3. Geef de branch een logische naam. Bijvoorbeeld `feature-about`.
4. VS Code schakelt daarna automatisch over naar deze nieuwe branch.
5. Alles wat je nu aanpast, doe je op deze branch en nog niet direct op `main`.

? Wisselen en mergen

Als je klaar bent met werken op je branch, kun je de branch samenvoegen met `main`. Controleer eerst of je wijzigingen zijn opgeslagen en gecommit.

1. Klik linksonder opnieuw op de branchnaam om te wisselen tussen branches.
2. Kies `main`. Je gaat nu terug naar de hoofdversie van je project.
3. Open de Command Palette met `Ctrl + Shift + P`.
4. Zoek en kies **Git: Merge Branch**.
5. Selecteer de branch die je wilt samenvoegen, bijvoorbeeld `feature-about`.
6. De wijzigingen uit die branch worden nu toegevoegd aan `main`.
7. Klik daarna op **Sync Changes** om de wijzigingen ook naar GitHub te sturen.

Let op: Gebruik branches zodra je aan een nieuwe feature begint. Zo blijft `main` netjes en stabiel.

? Overzicht: Git in VS Code (+ herhaling)

Actie	Hoe in VS Code?
Repo aanmaken	Source Control → Initialize Repository
Bestanden stagen	Klik op **+** naast een bestand of bij **Changes**
Committen	Typ een duidelijke commit message en klik op **Commit**
Pushen / pullen	Klik op **Sync Changes**
Publiceren naar GitHub	Klik op **Publish Branch**
Branch aanmaken	Klik op de branchnaam linksonder → **Create new branch**
Branch wisselen	Klik op de branchnaam linksonder → kies een branch
Branch mergen	Ga eerst naar `main` → `Ctrl + Shift + P` → **Git: Merge Branch**
Diff bekijken	Klik op een gewijzigd bestand in Source Control
Geschiedenis bekijken	Timeline of Git Graph

?? EINDOpdracht – Werken met branches

Maak een nieuwe feature op een aparte branch. In deze opdracht pas je de vormgeving van de website aan met CSS. Je maakt de achtergrond rood, de tekst wit en je past de titel van de pagina aan.

Daarna controleer je wat er gebeurt als je wisselt tussen `main` en je nieuwe branch. Je laat zien dat de code en de website er anders uitzien per branch. Voeg de branch daarna samen met `main`.

afbeelding.png

? Stappenplan

1. Maak een nieuwe branch met de naam `feature-red-theme`.
2. Controleer of VS Code nu op de branch `feature-red-theme` staat.
3. Open het CSS-bestand van je website. Bijvoorbeeld `style.css` of `css/style.css`.
4. Zorg ervoor dat de achtergrond van de pagina rood wordt.
5. Zorg ervoor dat de tekst op de pagina wit wordt.
6. Open het HTML-bestand van je website. Bijvoorbeeld `index.html`.
7. Pas de hoofdtitel (`<h1>`) aan naar een nieuwe titel, bijvoorbeeld `Mijn rode website`.
8. Controleer in de browser of de achtergrond rood is, de tekst wit is en de titel is aangepast.
9. Maak een screenshot van de website in de browser op de branch `feature-red-theme`. De rode achtergrond, witte tekst en aangepaste titel moeten zichtbaar zijn. Deze screenshot lever je later in.
10. Commit je wijziging via Source Control met een duidelijke commit message, bijvoorbeeld `Rode stijl toegevoegd`.
11. Wissel nu terug naar de branch `main`.
12. Controleer in de code of de rode achtergrond, witte tekst en aangepaste titel hier nog niet zichtbaar zijn.
13. Open of vernieuw de website in de browser.
14. Controleer dat de website op `main` nog de oude vormgeving en oude titel heeft.
15. Maak een screenshot van de website in de browser op de branch `main` vóór de merge. De oude vormgeving en oude titel moeten zichtbaar zijn. Deze screenshot lever je later in.
16. Wissel opnieuw terug naar de branch `feature-red-theme`.
17. Controleer in de code dat de aanpassingen weer zichtbaar zijn.
18. Vernieuw de website in de browser.
19. Controleer dat de rode achtergrond, witte tekst en aangepaste titel weer zichtbaar zijn.
20. Wissel daarna terug naar `main`.
21. Merge de branch `feature-red-theme` via **Git: Merge Branch**.
22. Klik op **Sync Changes**.

23. Controleer na de merge of de rode achtergrond, witte tekst en aangepaste titel nu ook op `main` zichtbaar zijn.
24. Maak een screenshot van VS Code waarin linksonder de branch `main` zichtbaar is na de merge. Zorg dat ook de commit-geschiedenis zichtbaar is, bijvoorbeeld via Timeline of Git Graph. Deze screenshot lever je later in.

Let op: Commit eerst je wijzigingen voordat je wisselt naar een andere branch. Anders kunnen je wijzigingen blijven “meeverhuizen” naar de andere branch. Dan zie je niet goed wat het verschil is tussen `main` en `feature-red-theme`.

? Inleveren

Lever de volgende 3 screenshots in:

1. Screenshot van de website in de browser op de branch `feature-red-theme`. De rode achtergrond, witte tekst en aangepaste titel moeten zichtbaar zijn.
2. Screenshot van de website in de browser op de branch `main` vóór de merge. De oude vormgeving en oude titel moeten zichtbaar zijn.
3. Screenshot van VS Code na de merge, waarin linksonder de branch `main` zichtbaar is. Zorg dat ook de commit-geschiedenis zichtbaar is, bijvoorbeeld via Timeline of Git Graph.
4. Een pdf waarin je uitlegt waarom het nuttig is om met branches te werken.

Revision #15
Created 2026-04-23 08:42:04 UTC by Aron
Updated 2026-07-02 09:12:57 UTC by Aron