



Inleiding en Praktische cursus

door Ali Kiyak en Lenny Kraaijenhof

Inhoudsopgave

INLEIDING	3
Wat is GIT? 3	
In deze cursus 3	
GIT VOOR TEAMS	4
Het belang van GIT 4	
<i>Een versiebeheersysteem, wat doet dat precies?</i>	<i>4</i>
<i>Samenwerken via GIT</i>	<i>4</i>
BASIS CONCEPTEN EN TERMINOLOGIE	5
Spreek je GIT? 5	
GIT VIA JE COMMAND LINE INTERFACE	7
Wat is een Command Line Interface? 7	
<i>Basis BASH commando's.....</i>	<i>7</i>
INSTALLATIE VAN GIT	8
Installatie op een Windows-omgeving 8	
1 De installatiebestanden downloaden.....	8
2 De installatie	8
3 Je identiteit instellen	9
Installatie op een MacOS- of Linux-omgeving 10	
AAN DE SLAG MET GIT.....	11
Een lokale <i>repository</i> maken11	
<i>Het init-commando</i>	<i>11</i>
Een wijziging toevoegen aan GIT via een <i>commit</i> 12	
<i>Het add-commando</i>	<i>12</i>
<i>Het commit-commando</i>	<i>12</i>
<i>Je toevoeging ongedaan maken</i>	<i>12</i>
Een <i>bare repository</i> aanmaken 12	
<i>Via het init-commando</i>	<i>12</i>
WERKEN MET REMOTES	13
1 Account aanmaken op Github.....	13
2 Remote repository aanmaken.....	13
3 Je remote toevoegen aan je lokale repository	14
Samenwerken via GIT 14	
<i>Het push-commando.....</i>	<i>14</i>
<i>Het pull-commando</i>	<i>14</i>
BESLUIT	16
Meer als gewoon de basis 16	
REFERENTIES	17
Extra informatie 17	
BIBLIOGRAFIE	18

Inleiding

Wat is GIT?



GIT is een versiebeheersysteem ontwikkeld door Linus Torvalds. Het wordt wereldwijd gebruikt en is bij uitstek het populairste systeem van zijn soort. Veel bedrijven en teams van ontwikkelaars zijn afhankelijk van een dergelijk systeem omdat het programmeren met anderen efficiënt en haalbaar maakt.

In deze cursus

GIT is handig voor iedereen die werkt in team, aan een groot project, met veel bestanden. Kortom programmeurs, systeembeheerders, noem maar op. Deze cursus geeft je een idee van het praktisch nut van GIT, en legt je uit hoe je er zelf mee aan de slag gaat.

Allereerst gaan we dieper in op de waarde van GIT voor een team. Wat het is, wat het doet en waarvoor het nuttig is. Daarna bespreken we de basis concepten en terminologie die komt kijken bij het gebruik van GIT.

Deze terminologie heb je nodig voor het werken met de verschillende commando's die GIT je ter beschikking stelt in de Command Line Interface (CLI). We geven je ook nog kort even mee hoe je je kunt behelpen in de CLI.

Tenslotte leren we je hoe je GIT installeert op diverse systemen, en hoe je de meest voorkomende acties uit voert. Deze acties zullen je natuurlijk voorbereiden om projecten te kunnen uitvoeren met GIT.

Het belang van GIT

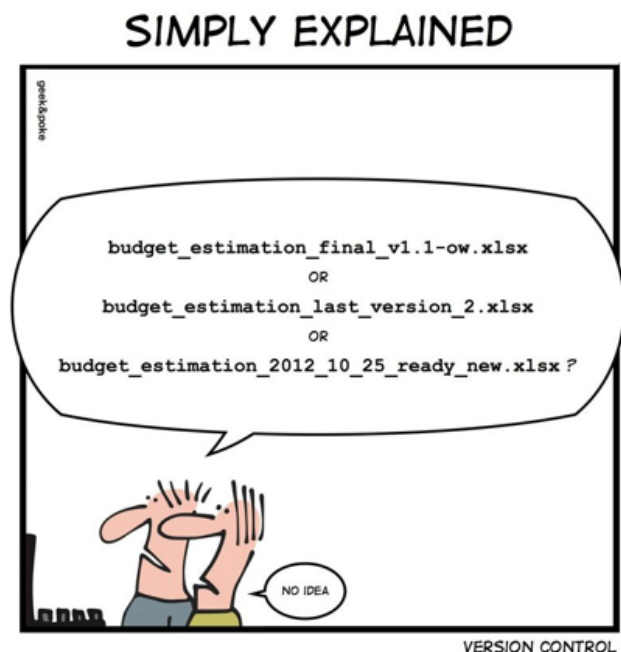
Een versiebeheersysteem, wat doet dat precies?

Een versiebeheersysteem houdt alle wijzigingen bij die ooit aan je project gemaakt werden. Maar het gaat verder dan dat, GIT houdt namelijk ook de volledige geschiedenis van een bestand tot op de regel bij. Dat betekent dat je precies kan achterhalen hoeveel regels code iemand heeft toegevoegd aan je project, wanneer die persoon dat gedaan heeft en om welke reden.

Samenwerken via GIT

Omdat GIT zo nauwkeurig bijhoudt wie, welke wijzigingen aan een bestand heeft gemaakt, kan GIT ook twee verschillende versies van een bestand samenvoegen. Dit noemen we een 'merge'.

Als een teamlid een regel code toevoegt aan een bestand, en een ander teamlid werkt op dat moment ook aan een functionaliteit in datzelfde bestand, dan is GIT intelligent genoeg om deze bestanden samen te voegen. Zo staat voor het volledige team altijd de meest recente versie van het project ter beschikking.



Weetje: Het verschil tussen Github en GIT

Misschien heb je ooit al eens iets gehoord over Github en vraag je je nu af of dat hetzelfde is als GIT. Goede vraag! Github is namelijk een bedrijf dat kan zorgen voor de hosting van je remote. Bovendien voegt de applicatie ook handige projectmanagement functionaliteiten toe zoals het bijhouden van issues, de projectvoortgang, pull-requests en meer.

Basis concepten en terminologie

Spreek je GIT?

Het is erg belangrijk om de basis concepten van GIT goed onder de knie te hebben. Alle GIT-commando's zijn hier namelijk op gebaseerd. Het is normaal dat je af en toe wat op moet zoeken, maar doe je best om ze zo goed mogelijk te begrijpen.

Repository

In een repository kan je de bestanden van je project terugvinden terugvinden. GIT wordt voornamelijk gebruikt om aan de broncode van software te werken, maar kan ook toegepast worden op andere bestanden. Sommige teams werken bijvoorbeeld ook samen aan online-documentatie via GIT.

Je maakt een repository door middel met de volgende instructie: `$ git init`

Working tree

De working tree is een map op je computer die in verbinding staat met een repository. In de working tree bevinden zich alle bestanden en mappen van je project. Je kan bestanden aanpassen, toevoegen en verwijderen zoals je dat normaal ook zou doen.

In dit stadium worden wijzigingen nog niet bijgehouden in GIT.

Bare repository & Remote

Een remote is een speciale soort repository namelijk een bare repository. Dit is een repository zonder working tree. Je kan dus niet rechtstreeks wijzigingen aanbrengen op deze repository.

De remote waakt over de laatste versie van je project en wordt door je hele team gebruikt als een soort middenman om de code uit te wisselen. Niemand werkt vanaf direct vanaf een remote, maar elk teamlid gebruikt de remote dus om alle laatste wijzigingen op te halen of te delen met de rest van het team.

Je maakt een remote repository door middel van de volgende instructie: `$ git init --bare`

Clone

Met behulp van het command `$ git clone <repository>` kan je de repositories van jezelf of van anderen kopiëren naar de map waar je je momenteel bevindt. De repositories kunnen ergens lokaal staan (ergens op je eigen computer) of op het internet.

Je hoeft geen `$ git init` meer uit te voeren, de working tree is namelijk direct beschikbaar.

Staging tree

In de staging tree bevinden zich alle aanpassingen die je maakte nadat je ze had toegevoegd met behulp van het `$ git add <bestandsnaam>` commando.

Commit

Een commit registreert alle wijzigingen in bestanden die momenteel in je staging tree staan bij GIT. Alle wijzigingen worden opgeslagen in de repository. Je geeft bij een commit altijd aan

waarom je de wijziging hebt gemaakt. Een commit gebeurt via het volgende commando `$ git commit -m <een kort bericht>`

Push

Met het `$ git push` commando, kan je de wijzigingen die zijn gebeurd van lokale repository ook uploaden naar een remote repository. Zo hebben je teamleden en andere altijd de laatste versie van het project ter beschikking.

Pull

Een pull haalt juist nieuwe gegevens op van een remote. Wanneer een van je teamleden een wijziging pusht, kan jij deze ophalen met het commando: `$ git pull`.

Merge

Een merge vindt plaats wanneer twee verschillende nieuwe versies van het project bij elkaar gevoegd worden. Wanneer je teamlid bijvoorbeeld aan hetzelfde document gewerkt heeft als jij, kan een merge plaatsvinden. Er zijn twee nieuwe versies van hetzelfde bestand.

Conflict

Soms kan het zijn dat GIT niet automatisch een merge kan uitvoeren. Twee teamleden hebben bijvoorbeeld aan dezelfde regel code gewerkt. Als een conflict plaatsvindt, moet jij zelf bepalen welke regel code blijft staan.

Weetje: Bestanden negeren met gitignore

Soms wil je dat bepaalde bestanden die je in je lokale repository hebt staan, niet toegevoegd kunnen worden aan GIT. Dat kan bijvoorbeeld handig zijn voor bestanden die tijdens het programmeren gegenereerd worden, of wanneer jouw versie van een stuk code anders moet zijn dan die van je collega's. Wanneer je zo'n bestand hebt gemaakt, voeg je de locatie met bestandsnaam toe aan een file ``.gitignore``. Nu zal hij niet meer in je staging tree komen.

Wat is een Command Line Interface?

Meeste besturingssystemen maken gebruik van een Graphical User Interface, oftewel een GUI. Een GUI toont afbeeldingen, schermen, geformatteerde teksten om interactie met de gebruiker makkelijker te maken. Een GUI is handig (en ook beschikbaar) voor verschillende doeleinden, maar voor versiebeheer met GIT leer je best eerst de CLI-versie kennen.

Een Command Line Interface, afgekort CLI, is een tekst-gebaseerde interface. Via de CLI geef je de computer instructies met behulp van commando's. Commando's kunnen zeer simpel zijn (bijvoorbeeld de inhoud van een map bekijken), maar ze kunnen ook zeer complex en krachtig worden naargelang je een ervaren gebruiker wordt.

GIT werd oorspronkelijk geschreven voor BASH, de Bourne Again Shell. BASH is een UNIX shell. UNIX is een oud besturingssysteem waarop o.a. Linux en MacOS gebaseerd zijn. Wanneer je werkt vanaf een Linux of MacOS besturingssysteem kan je GIT simpelweg gebruiken vanaf je terminal. Met een Windows-pc ga je anders te werk.

Werk je met Windows dan ga je voor deze cursus 'GIT Bash' nodig hebben. Dit is een programma dat GIT en een gedeelte van Bash beschikbaar maken op Windows.

Basis BASH commando's

Om je op weg te helpen geven we je graag een paar basis commando's van de Bash-shell mee:

- | | |
|---|---|
| ⇒ <code>\$ cd <relatief-of-absoluut-pad></code> | <u>Change Directory</u> : Van map wisselen |
| ⇒ <code>\$ ls</code> | <u>List</u> : De inhoud van een map bekijken |
| ⇒ <code>\$ mv <oorsprong> <doel></code> | <u>Move</u> : Bestanden en mappen verplaatsen |
| ⇒ <code>\$ cp <bron> <doel></code> | <u>Copy</u> : Bestanden en mappen kopiëren |
| ⇒ <code>\$ mkdir <naam></code> | <u>Make Directory</u> : Map aanmaken |

Er zijn nog veel meer commando's. Wanneer je de CLI vaak genoeg gebruikt word je er vanzelf wegwijs mee. Nu focussen wij ons in deze cursus verder op de specifieke GIT-commando's.

Installatie van GIT

Installatie op een Windows-omgeving

1 De installatiebestanden downloaden

Om GIT commands te kunnen uitvoeren moet je eerst GIT installeren op je computer. Je doet dit door eerst naar de officiële website van GIT te gaan, namelijk: <https://git-scm.com/>.

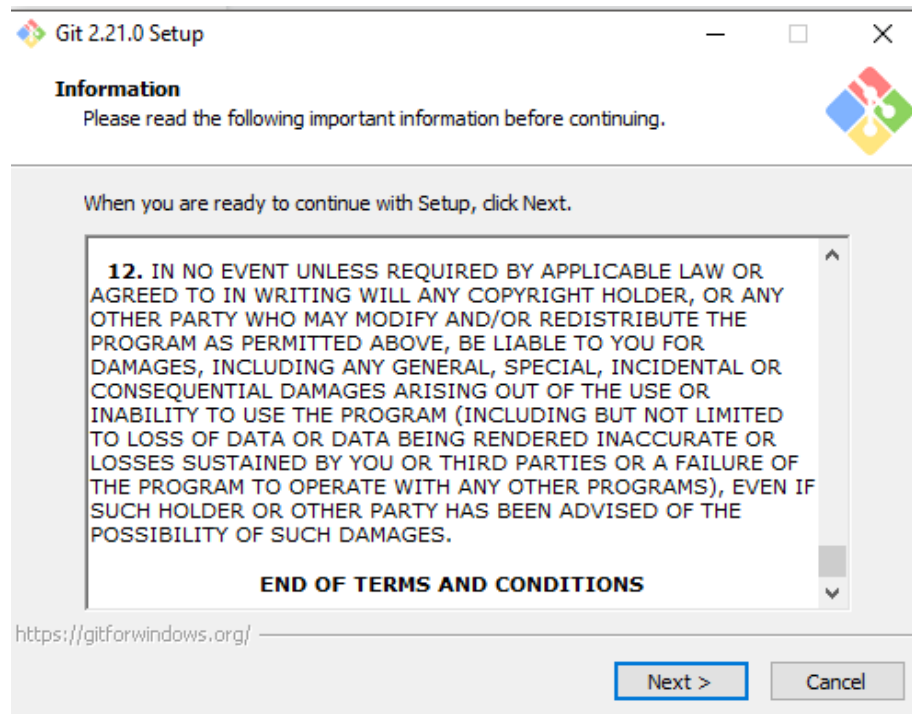


Je downloadt eerst de laatste versie van GIT. Zo heb je altijd de meest stabiele versie.

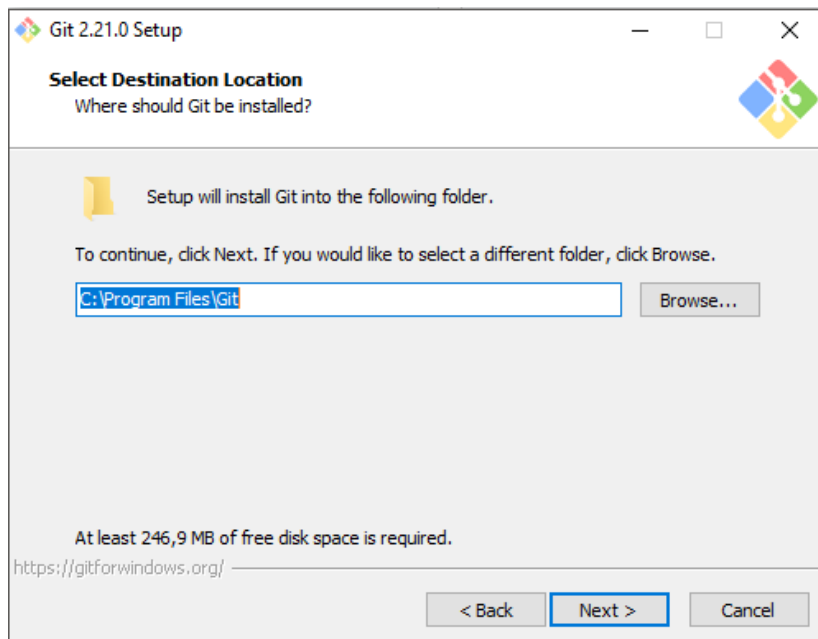
2 De installatie

Nadat de laatste versie is gedownload, start je het installatiebestand. De standaard installatie-instellingen van GIT zijn perfect voor ons, maar we overlopen de belangrijkste hieronder zodat je begrijpt wat er gebeurt.

Eerst krijg je een melding met de 'terms and conditions' van GIT en de licence die het gebruikt. GIT is compleet gratis en de broncode van de software is open-source, dat betekent dat jij ze kan bekijken en zelfs aanpassen.



Je kiest hierna waar je GIT willen installeren op de computer. Je kan de locatie aanpassen naar eigen keuze maar wij raden je toch aan om de standaardlocatie op de computer te gebruiken.

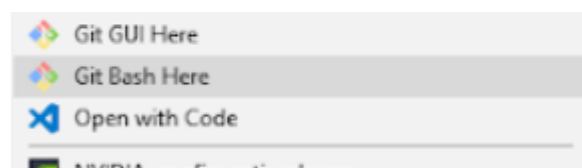


Je kiest hierna de gewenste individuele componenten van GIT. De standaard aangeduide componenten zijn goed en aanpassingen zijn niet nodig.

Vervolgens kan je gewoon de installatie versnellen door op 'Next' te drukken en vervolgens de installatieprocedure af te sluiten.

3 Je identiteit instellen

Nadat de installatie gedaan is klik je met je rechtermuisknop ergens op het bureaublad, en selecteer je 'Git Bash Here'. Een Bash-venster opent nu.

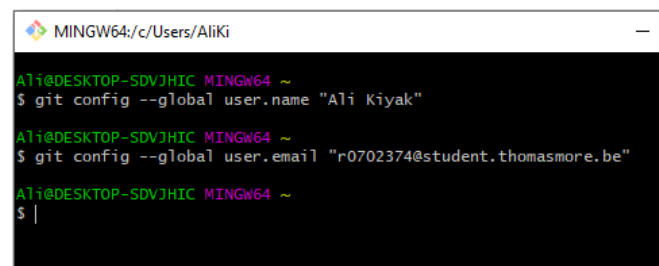


Je gaat daarna twee commando's uitvoeren om de installatie van GIT te vervolledigen.

Voer de volgende commando's achtereenvolgens uit:

```
$ git config --global user.name "voornaam naam"
$ git config --global user.email "e-mail"
```

De installatie van GIT op je computer is nu gebeurt!



Installatie op een MacOS- of Linux-omgeving

Gebruik gewoonweg je favoriete Package Manager en, afhankelijk van je besturingsysteem, gebruik je een van de volgende commando's:

MacOS, met Brew: `$ brew install git`

Ubuntu, met apt-get: `$ sudo apt-get install git`

Je gaat daarna twee commando's uitvoeren om de installatie van GIT te vervolledigen.

Voer de volgende commando's achtereenvolgens uit:

```
$ git config --global user.name "voornaam naam"
```

```
$ git config --global user.email "e-mail"
```

Aan de slag met GIT

Een lokale *repository* maken

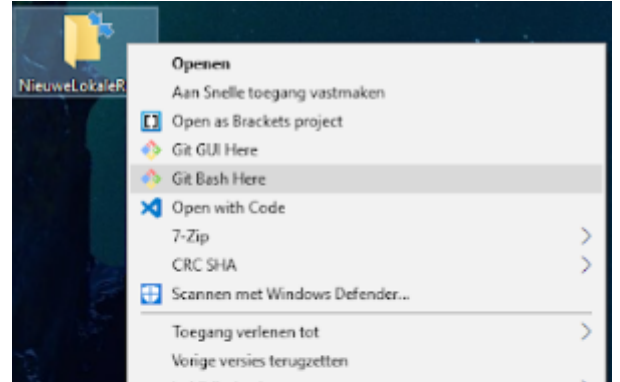
Het *init*-commando

In een repository kan je de bestanden van je project terugvinden. GIT wordt voornamelijk gebruikt om aan de broncode van software te werken, maar kan ook toegepast worden op andere bestanden. Sommige teams werken bijvoorbeeld ook samen aan online-documentatie via GIT.

Om een lokale repository te maken moet je een GIT bash op je gewenste directory openen.

Je voert de command `$ git init` uit. De initialiseert dan een lokale repository op de directory waar het bash venster zich momenteel bevindt.

De lokale repo is na het uitvoeren van deze command gemaakt.



Een wijziging toevoegen aan GIT via een commit

Het *add*-commando

Als je een wijziging hebt gemaakt binnen je working tree, voeg deze dan eerst toe aan je staging tree. Alle wijzigingen die je toevoegt, zullen onderdeel worden van 1 commit.

Gebruik `$ git add .` om alle wijzigingen toe te voegen aan je staging tree.

Gebruik `$ git add <pad>` om een specifiek bestand toe te voegen aan je staging tree.

Het *commit*-commando

Wanneer je verschillende wijzigingen hebt gemaakt, en deze in je staging tree staan kan je een commit maken. Dit plaatst de wijzigingen officieel in je repository. Zorg er voor dat de wijzigingen die je maakte op een logische manier bij elkaar passen. De vuistregel is de volgende, je moet in 1 duidelijke regel kunnen omschrijven waarom de bestanden in je staging tree werden aangepast.

Gebruik `$ git commit` om je commit te maken. Een teksteditor verschijnt om een lange beschrijving toe te voegen.

Gebruik `$ git commit -m "Je korte omschrijving."`. Nu is je korte omschrijving voldoende.

Je toevoeging ongedaan maken

Maak je commit ongedaan met het commando: `$ git reset --soft HEAD~1`.

Nog verder terug, dat kan ook. Vervang `'HEAD~1'` met bijvoorbeeld: `HEAD~2`.

Een *bare repository* aanmaken

Via het *init*-commando

Je kan zelf een remote aanmaken d.m.v. het init-commando. Een remote is niks anders dan een repository zonder working tree, ook wel een *bare repository* genoemd.

Gebruik hiervoor het volgende commando: `$ git --bare init`

Remotes toevoegen, verwijderen en bekijken

Nu je zelf een lokale repository hebt gemaakt gaan we deze online plaatsen zodat deze beschikbaar is voor je teamleden en dat je ook een back-up hebt van je project.

Je gaat hiervoor gebruik maken van GitHub. Github bied je gratis hosting aan voor je remote.

1 Account aanmaken op Github


Als je nog geen Github-account hebt, ga dan eerst naar de GitHub 'Sign-Up'-pagina:

<https://github.com/join>

Maak een account aan door je informatie in te geven. Voor deze cursus kies je de gratis optie die GitHub aanbiedt. Je kan later nog veranderen naar een betaalde versie als je wilt.

2 Remote repository aanmaken

Nadat het account is gemaakt, en geverifieerd, ga je een repository maken.

Klik op de groene  **New** knop om je repository aan te maken.

Daarna ga je de instellingen van de remote repository ingeven.

Kies een naam

Owner

Repository name *

AliKiyak / repository

Great repository names are short and memorable. Need inspiration? How about **cuddly-octo-waddle**?

Schrijf een omschrijving

Description (optional)

Een beschrijving voor de repo

Kies voor een Privé repository

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

☒ Private
You choose who can see and commit to this repository.

Maak nog GEEN readme-file aan.

☐ 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.

Opslaan

Add .gitignore: None

Add a license: None

Create repository

De naam van je repository

Eerst geef je de naam in van de remote repository. Deze moet uniek zijn tussen al jouw repositories.

De omschrijving van je repository

Daarna kan je ook een beetje uitleg meegeven over de remote repository die je maakt.

Publiek of privé?

Je kan dan kiezen of uw remote repository *public* of *private* word. Een *public repository* is beschikbaar over heel het internet (niet alleen voor degene die een GitHub account hebben). Deze optie gebruik je als je je project wilt delen met de wereld en NIET voor je privé-projecten.

Een *private repository* kan alleen jij en de personen die je hebt toegevoegd aan het project zien. Deze optie pak je als je aan projecten werkt die alleen jij of jouw team mag bekijken en aanpassen. Wij raden aan om altijd een private repository aan te maken tenzij je een open-source-project start.

3 Je remote toevoegen aan je lokale repository

Nu linken we je online Github-repository aan je lokale repository. Hiervoor voeg je de Github repository toe als remote. Gebruik hiervoor het volgende commando:

```
$ git remote add origin https://github.com/username/remotereponaam.git
```

Je vervangt 'username' en 'remotereponaam.git' natuurlijk met je eigen instellingen.

Samenwerken via GIT

Het *push*-commando

Je hebt nu een lokale repository met een working tree, en een remote waarop je je project online kan bewaren. Om je remote een 'update' te geven, gebruik je het *push*-commando. Dat doe je als volgt:

```
$ git push origin master
```

Hierbij staat origin voor naam van je remote, en master voor de branch van die remote. De naam van je remote hebben we in het vorige hoofdstuk traditioneel 'origin' genoemd. De branch 'master' is een standaard branch, maar je hoeft je daar niks van aan te trekken voor deze cursus. Werken met branches is voor gevorderden.

Het *pull*-commando

Omgekeerd kan jouw lokale repository ook niet meer up-to-date zijn met de remote. In dit geval wil je dus de nieuwe versie van je project ophalen, dat doe je met het *pull*-commando. Je gebruikt 'm als volgt:

```
$ git pull origin master
```

Conflicten oplossen

Soms ontstaan er conflicten tijdens een pull of push. Dit gebeurt wanneer Git de twee verschillende versies van je repository niet automatisch kan mergen (mengen). Je moet dan manueel het bestand getroffen aanpassen tot het klopt, en een commit uitvoeren.

Daarna is de pull of push compleet.

Besluit

Meer als gewoon de basis

Het lijkt misschien niet zo, maar je kan nu al veel met GIT. Je kan repositories aanmaken, je kan aanpassingen doorgeven aan je repositories en je kan met een remote werken. Dit is niet alleen de basis van GIT, maar de bouwstenen waar het op is gebouwd.

GIT lijkt misschien niet handig in het begin, maar je zal erachter komen dat GIT één van handigste hulpmiddelen is die een softwareontwikkelaar kan hebben.

In je toekomst ga je werken aan verschillende projecten. Dit kunnen softwarematige projecten zijn, maar ook verslagen, rapporten die je alleen of in team gaat maken. We hopen dat met de kennis die je hebt geleerd, dat je voor deze projecten GIT zult gebruiken.

Referenties

Extra informatie

Documentatie van GIT

De documentatie van GIT zelf bevat alle commands die je kan gebruiken met een uitgebreide uitleg. Je kan nu al veel met GIT, maar als je je zelf toch wilt verdiepen in GIT dan is de documentatie een goed startpunt.

Je kan de documentatie van GIT hier terugvinden: <https://git-scm.com/docs>.

Pro Git book

De Pro Git boek was geschreven door Scott Chacon en Ben Straub. In dit boek gaan dieper in de mogelijkheden van GIT. Ze doen dit met voorbeelden, scenario's en veel meer. Het boek is helemaal gratis online te downloaden.

Je kan het boek hier terugvinden: <https://progit2.s3.amazonaws.com/en/2016-03-22-f3531/progit-en.1084.pdf>

Learn Git van Codecademy

Op Codecademy kan je de basis van GIT nog eens gratis overlopen. Ze doen met tekstbestanden en CLI die ze hebben geïmplementeerd in de browser. Zo kan je op een interactieve manier je kennis van GIT herhalen.

Je kan de cursus hier terugvinden: <https://www.codecademy.com/learn/learn-git> .

Best practices

Iedereen gebruikt GIT op zijn eigen manier, maar er zijn toch wel wat richtlijnen waar iedereen zich aan moeten kunnen houden.

Je kan een goede samenvatting van deze richtlijnen terugvinden op:
<https://gist.github.com/pandeiro/1552496>

Bibliografie

Je kan een enorme hoeveelheid kennis terugvinden over de CLI, GIT en Github terugvinden op het internet. Toen wij deze documentatie opmaakte, hebben wij gebruik gemaakt van volgende bronnen:

Chacon, S., & Straub, B. (2014).

Pro Git Book. Opgehaald van git-scm.com: <https://git-scm.com/book/en/v2>

Computer Hope. (13 November 2018).

Command Line. Opgehaald van computerhope.com:
<https://www.computerhope.com/jargon/c/commandi.htm>

GIT. (z.d).

Git Documentation. Opgehaald van git-scm.com: <https://git-scm.com/docs>

Github. (z.d).

Github Guides. Opgehaald van guides.github.com: <https://guides.github.com/>