VHT: Git

2013-2014

# http://blog.eogn.com/.a/6a00d8341c767353ef016762f7c808970b-800wiWerkwijze

Tijdens de komende twee VHT’s ga je leren werken met git.   
Git zorgt er voor dat je met een team kunt werken aan een project, terwijl je allemaal met dezelfde folders en bestanden bezig bent. Er zijn van Git twee aanvullende vaardigheidstrainingen die tijdens dit project worden gegeven om jullie hier in te ondersteunen.

In feite werk je dus met zijn allen in een centrale projectfolder, die op een Git-server staat opgeslagen. Ieder teamlid kopieert die gegevens lokaal naar zijn/haar laptop, zodat naast de centrale projectfolder, iedereen hier een kopie van heeft. Het is dan van belang dat iedereen toegang heeft tot die Git-server. Daarom werk je in VHT2 met de clouddienst Github.com.

## VHT 1

De eerste VHT laat jou kennis maken met het installeren en configureren van GIT. Hierbij leer je de basic command line ‘git commands’ die worden gebruikt tijdens het werken in een project.

* Werken met git in de command line
* Individueel lokaal werken op je computer

## VHT 2

In VHT 2 ga je aan de slag met een groep van 4 personen, waarbij je in github een gezamenlijk project aangaat. Ieder draagt bij aan het project door middel van git commits.

* Werken met git dmv Github for Windows/ Github for Mac
* Vanaf de gitservers van github werken met meerdere projectleden

# VHT 1

Installeren GIT op PC.

<http://msysgit.github.com/>

*Testen van GIT*

- command line prompt openen

      1. open de betreffende folder in je windows verkenner

       2. selecteer de url balk en typ 'cmd'

- git bash prompt openen

    1. rechtermuisknop in de folder bij windows verkenner

    2. selecteer git bash

    voer in de command line in: **git --version**

**1. welke versie zie je verschijnen in de command line?**

*Configureren van GIT*

**git config:**

         Eerste wat je doet na installeren van git op je computer is het declareren van je gebruikersnaam en email adres. Dit is belangrijk omdat iedere git 'commit'(komen we later op) deze informatie gebruikt.

open de command line (command prompt of git bash)

typ: **git config** --global user.name "Jouw Username"

waarin 'jouw username' uiteraard echt jouw username betreft.

**2. Wat betekent de --global die je hebt toegevoegd aan je command?**

**3. Welke 2 andere invoersoorten naast --global kun je gebruiken, en wat is het verschil tussen die drie?**

Voer dezelfde command uit, maar dan om je email-adres te declareren

**4. Welke git-opdracht heb je gebruikt om de email adres te declareren?**

Je hebt nu de meest algemene (en noodzakelijke) configuraties gedaan. Tijd om te gaan spelen met git!

*Eerste stapjes met GIT*

De eerste stap in het werken met git is er voor zorgen dat je een repository maakt.

**5. Leg in eigen woorden uit wat een repository is.**

Met de command opdracht '**git init** <naam folder>' maak je een folder op je computer die gelijk 'git-proof' is. Dat wil zeggen dat deze folder direct al een repository is, lokaal op je computer. Daar kan nu nog niemand bij, en aangezien we git juist graag gebruiken om samen te werken, zullen we in de volgende VHT laten zien hoe je zo een repository in de 'cloud' aanmaakt.

Open je code-editor. In de folder die je zojuist heb geinitialiseerd maak je een bestand aan genaamd index.html.

Maak daarin een standaard html opzet. zet in de body wat inhoud, bijvoorbeeld:

<h1> GIT Training voor hardcore Radius Developers! </h1>

We hebben nu een 'change' gemaakt in onze folder, die we vanaf nu onze repository blijven noemen.

open je command line, en navigeer naar je repo.

voer de opdracht uit '**git status**'  uit.

Je ziet nu in rood staan: **modified:    index.html**

Rood betekent dat er een bestand is gewijzigd, maar dat deze nog niet in de 'staging' fase staat van de commit.

Met de invoer '**git add'**<bestandsnaam> voeg je het bestand toe in de 'staging' fase.

**6. Leg in eigen woorden uit wat de staging fase is in git**

Nu je het bestand in de 'staging' area gezet hebt gaan we nog eens de status van onze repo bekijken.

**7. bekijk nogmaals de status van de repo. Wat zijn de verschillen tussen de vorige status-check en deze status-check?**

Bestanden die ge-'staged' zijn, staan klaar om gecommit te worden. de commit is de daadwerkelijke change die jij gaat doorvoeren in de repository. dat betekent dat er achter de schermen als het ware een 'foto' wordt gemaakt van de nieuwe situatie, en we altijd nog naar terug kunnen naar de oude 'foto' als we bijvoorbeeld iets fouts hebben gedaan.

voer de opdracht: **git commit** -m "toevoegen van index.html"   uit.

**8. Leg uit wat je nu zojuist met deze invoer gedaan hebt?**

voer de opdracht uit **git log**

**9. Beschrijf de informatie die je terugkrijgt van de command-line**

Oeps! het was helemaal niet de bedoeling om deze index.html file in het project toe te voegen. Wat je nu gaat doen is dat je de laatste commit gaat terugdraaien. Een belangrijk aspect hierbij is dat je niet letterlijk terug de 'geschiedenis' in gaat en die gaat herschrijven, maar

voer de opdracht uit  **git reset** --hard HEAD~1

**10. Leg uit wat je met deze invoer hebt gedaan:**

**- wat betekent de invoer git reset?**

**- wat betekent --hard?**

**- wat wordt bedoeld met HEAD?**

**- wat wordt bedoeld HEAD~1?**

Omdat herhaling op verschillende manieren helpt om je beter te laten begrijpen wat je doet, ga je nu naar de volgende link:

*begeleider: neem ze gezamenlijk mee in de eerste 3 'challenges' zodat je ze uitleg kan geven over hoe het programma werkt en waar ze moeten kijken*

<http://try.github.io/levels/1/challenges/1>

hierbij kom je weer door de dingen die je zojuist hebt gedaan. Heb je deze vht gevolgd en 'Try git' gedaan, dan heb je een goede basis weggelegd voor GIT.

In de volgende VHT gaan we verder met github  en gaan we in een projectgroepje samen aan de slag om mbv van git en github samen te werken aan een project.