# Module Software Development

## Analyseren (week 3)

## Opdracht 1: git installeren en eenvoudige git-operaties.

Uiterste inleverdatum **8-10-2017.**

Inleveren **uitsluitend** via **Blackboard.** Inleveren **uitsluitend** als **.docx** of **.pdf**.

Naam en klas vermelden in het document!

Deze opdrachten gaan over versiebeheer. Meer specifiek: over het werken met git.

Deze opdracht moet je individueel inleveren (bij het werk mag je natuurlijk met anderen samenwerken).

## Wat moet je inleveren?

Uiteindelijk lever je individueel één (Word- of .pdf) document in met:

De gevraagde uitvoer 1 t/m 6.

Het spreekt voor zich dat je je naam en klas ook in het document vermeldt. Probeer ook te letten op leesbaarheid en lay-out.

## Documentatie

In de presentaties staan veel commando’s kort uitgelegd maar vaak heb je meer informatie nodig.

De officiele git-documentatie is te vinden op <https://git-scm.com/documentation>. Op Blackboard staat onder meer een git-cheat sheet document.

Op het internet is nog veel meer documentatie hierover te vinden. Zoek dit op indien nodig.

Zie ook hieronder onder ‘veel voorkomende acties’.

## Veel voorkomende acties

(shell commando’s, uit Unix/Linux, ook beschikbaar in git bash)

*Controleren inhoud working directory:*

ls

of liever (geeft meer informatie)

ls –l

(git commando’s)

*Controleren en overzicht commits, uitzoeken hashcodes:*

git log

*Controleren toestand repository, working directory en staging area:*

git status

Stap 1: installeren git.

Ga naar <https://git-scm.com/download> en download de juiste versie van git voor jouw systeem. Installeer git.

Voor Windows: zorg ervoor dat de installer icoontjes aanmaakt op je desktop en druk daarna 2 keer op volgende. Selecteer de optie “use git from windows command prompt” en druk 2 keer op volgende en wacht totdat git is geinstalleerd. Dit kan enkele minuten duren. Vink view releasenotes.rtf uit en druk op klaar.

Voor dat je git echt kan gebruiken moet je na de installatie eerst een email-adres en een gebruikersnaam invoeren. Anders kan je namelijk geen commits doen en krijg je daarbij een foutmelding.

Dit doe je met de volgende commando’s:

git config --global user.email ”*your.email@comes.here”*

git config --global user.name ”*your user name*”

Bijvoorbeeld:

git config --global user.email ”*stud1700@student.nhl.nl”*

git config --global user.name ”StudentHBO-ICT!”

Stap 2: Aanmaken lokale repository.

Open de git-commando omgeving, git bash (op windows) of terminal (op Mac/Unix/Linux).

**NB: In het vervolg zal de term ‘shell’ gebruikt worden voor de commando-omgeving.**

Maak een directory aan waarin je straks gaat werken en geef deze een geschikte naam (bv: GitHuiswerk).

Ga nu naar deze directory en maak een nieuwe git repository aan..

Controleer dat deze inderdaad aangemaakt is.

**UITVOER 1:**

Maak een screenshot van de shell met de resultaten van deze stap.

Laat alle uitgevoerde commando’s zien, plus de uitvoer in de shell.

Zorg dat het screenshot goed leesbaar is!

Stap 3: Maken van een aantal bestanden

Maak in de git working directory drie nieuwe tekstbestanden aan. Noem die bijvoorbeeld anne1.txt, anne2.txt en anne3.txt (als je anne heet). Zet in elk bestand een eenvoudige tekst (Bijvoorbeeld: Hallo, dit is versie 1 van anne1.txt) en sla op.

**N.B.** Tekstbestanden kan je op verschillende manieren aanmaken en bewerken. Op een windows-systeem kan je Notepad (ook wel Kladblok genoemd) gebruiken. Maar een programmeur gebruikt liever een fatsoenlijke -gratis- editor, bijvoorbeeld Notepad++ (<https://notepad-plus-plus.org/>). Ook voor mac-systemen zijn er goede alternatieven voor de systeem-teksteditor. Je kan ook de in de shell ingebouwde editor vim gebruiken, hier een link naar [documentatie](http://vimdoc.sourceforge.net/) daarover.

Stap 4: Toevoegen aan de git repository

Voeg de bij stap 3 aangemaakte bestanden toe aan de staging area en controleer dat deze inderdaad nu bij git bekend zijn.

Stap 5: Eerste versie: versie 0.1

Sla deze bestanden op in je als versie 0.1.

Controleer ook dat dit gelukt is.

**UITVOER 2:**

Maak een screenshot van de shell waarin stap 3 t/m 5 te zien zijn.

Laat alle uitgevoerde commando’s zien, plus de uitvoer van de shell.

Stap 6: Weggooien en terughalen

Verwijder nu een van je bestanden. Controleer dat het bestand inderdaad weg is. Haal het bestand nu weer terug met git en controleer dat het inderdaad weer terug is.

**UITVOER 3:**

Maak een screenshot van de shell waarin stap 6 te zien is, tot en met het terughalen van het bestand.

Laat alle uitgevoerde commando’s zien, plus de uitvoer van de shell.

Stap 7: Bewerken van bestand

Bewerk een van je bestanden door er een extra regel aan toe te voegen. Voeg dit bestand aan de staging area toe.

Stap 8: Nieuw bestand

Maak ook een nieuw tekstbestand. Voeg ook dit toe aan de staging area.

Stap 9: Nieuwe versie

Maak een nieuwe versie (versie 0.2) waarmee de wijzigingen uit stap 7 en 8 worden gecommit. Controleer dat deze versie is opgeslagen.

**UITVOER 4:**

Maak een screenshot van de shell waarin stap 7 t/m 9 te zien is.

Laat alle uitgevoerde commando’s zien, plus de uitvoer van de shell.

Stap 10: zet vorige versie terug

We gaan nu de vorige versie (versie 0.1) terugzetten. Laat eerst zien wat er nu in je working directory staat. Bepaal daarna de zogeheten hash-code[[1]](#footnote-1) voor versie 0.1.

Haal nu versie 0.1 weer terug (daarvoor gebruik je de eerste 7 cijfers van de hash-code met git checkout).

Verifieer dat nu de vorige versie in je working directory is teruggezet. Bekijk tenslotte ook het bij stap 7 bewerkte bestand om te controleren dat je nu weer de vorige versie van dat bestand hebt, de eenvoudigste manier daarvoor is het commando cat *filename*.

**UITVOER 5:**

Maak een screenshot van de shell waarin stap 10 te zien is.

Laat alle uitgevoerde commando’s zien, plus de uitvoer van de shell.

Stap 11: ga weer naar de laatste versie

Tenslotte zetten we de laatste versie (versie 0.2) weer terug. Het commando is

git checkout master.[[2]](#footnote-2)

Verifieer dat nu de laatste (nieuwste) versie weer in je working directory is gezet. Bekijk tenslotte ook het bij stap 7 bewerkte bestand om te controleren dat je nu weer de nieuwe versie van dat bestand hebt.

**UITVOER 6:**

Maak een screenshot van de shell waarin stap 11 te zien is.

Laat alle uitgevoerde commando’s zien, plus de uitvoer van de shell.

1. De hash-code is een hexadecimaal getal als 765711a73970fe358c599303771b546defb35b5b.

   Gebruik git log om de juiste hash-code te achterhalen. Je hoeft gelukkig alleen de eerste 7 tekens te typen. [↑](#footnote-ref-1)
2. Onder windows kan het zijn dat je een melding als “error: cannot stat [filename]: permission denied” krijgt. Dit komt door een bug in windows. Sluit alle vensters, start git bash opnieuw op en probeer het opnieuw. Waarschijnlijk lukt het nu wel. Zo niet, reboot dan de computer. [↑](#footnote-ref-2)