# Git- ja GitHub-perusteet

## Tehtävän kuvaus

Tehtävän tarkoitus on tutustuttaa git:n peruskäyttöön ja varmistaa, että asennukset ja työkalut ovat kunnossa ensi kerran harjoituksia varten. Jos olet varma, että ne ovat jo sinulla kunnossa, perustele, miksi voit jättää tutoriaalin tekemättä.

Koska haarautuminen ei ole minulle entuudestaan tuttu, niin tein tämän tehtävän kokonaan.

## Gitin repository

Gitin tarkoituksena on säilöä muokattavaa lähdekoodia, jonka avulla voidaan palauttaa aikaisempia versioita, kehittää eri versioita yhtä aikaa hallitusti ja samalla muutoksia voidaan seurata, dokumentoida ja hallita. Git tallentaa tiedot paikalliseen tietovarastoon eli repositoryyn. Sen sisältöä voidaan synkronoida pilvessä olevaan erilliseen etä-repositoryyn tarpeen vaatiessa. Pilven avulla voidaan koodin kehitystyötä tehdä yhdessä tuvallisesti muiden kehittäjien kanssa.

## Gitin konfigurointi

Gitin asennuksen jälkeen käyttäjän tulee yhden kerran asettaa Gitin globaalit käyttäjätiedot kohdalleen:

$ git config --global user.name "Leo Sutinen"

$ git config --global user.email leo.sutinen@myy.haaga-helia.fi

Yksittäisen repositoryn käyttäjätietoja voidaan muokata, kunhan repository on ensin luotu.

$ git config user.name "Leo Sutinen"

$ git config user.email leo.sutinen@myy.haaga-helia.fi

Git käyttää oletuseditorina vi-editoria, jonka voi halutessaan muuttaa käyttämään esim. Visual Studio Codea.

$ git config --global core.editor "code --wait"

Käyttäjän halutessa voidaan valita myös yksittäisen git-projektin editoriksi "code":

$ git config core.editor "code --wait"

Gitin yleisellä tasolla olevia asetuksia voi tutkia list-option avulla:

$ git config --list

## Gitin peruskäyttö

Repositoryn voi perustaa käyttämällä hakemistoa, jota vielä ei ole otettu käyttöön versionhallinnassa.

$ mkdir 2023OhjProj1

$ cd 2023OhjProj1

$ git init

Init-komennon jälkeen hakemistoon syntyy .git-hakemisto, jonne Gitin konfigurointitiedot tallentuvat.

Mikäli repository on jo olemassa esim. pilvipalvelussa, se voidaan ottaa käyttöön clone-komennolla. Tämä on kuvattu etä-repositoryn (remote) kohdalla.

$ git clone <https://github.com/leksu70/2023OhjProj1.git>

Komento luo uuden alihakemiston 2023OhjProj1-nimellä, joka sisältää Gitin konfigurointitiedot .git-hakemistossa ja se kopioi alkuperäisen repositoryn tiedot ja viimeisimmän version luotuun alihakemistoon. Samalla luotu uusi paikallinen repository lisätään alkuperäisen repositoryn etärepositoryksi.

## Git-hallinta

Tiedostoja voidaan viedä Gitin hallintaan add-komennolla ja samalla voidaan myös lisätä kaikki hakemiston alla olevat tiedostot ja hakemistot hallintaan.

$ git add testitiedosto.txt

$ git add . 🡨 Kaikki tiedostot lisätään git:iin.

Lisättyjen tiedostojen tila voidaan tarkastaa status-komennon avulla:

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: testitiedosto.txt

Muutoksia hallinnoidaan commit-komennolla.

$ git commit🡨 Avaa oletusediton kommenttia varten.

$ git commit -m "Lisätty ominaisuus XXX"

[master (root-commit) 526996c] Lisätty ominaisuus testi

1 file changed, 1 insertion(+)

create mode 100644 testitiedosto.txt

Tiedostoihin tehdyt muutokset (commit) näkyvät commit-historian log-komennolla.

$ git log

commit 526996cae4abd850d56d45592da341a1b53b89d8 (HEAD -> master)

Author: Leo Sutinen <leo.sutinen@myy.haaga-helia.fi>

Date: Sun Jan 22 16:48:13 2023 +0200

Lisätty ominaisuus testi

Muutokset edelliseen versioon voidaan tutkia diff-komennolla.

$ git diff testitiedosto.txt

diff --git a/testitiedosto.txt b/testitiedosto.txt

index 5a192f0..8f4bcf1 100644

--- a/testitiedosto.txt

+++ b/testitiedosto.txt

@@ -1 +1,2 @@

-Tämä on testitiedosto, joka on luotu git init:n jälkeen.

\ No newline at end of file

+Tämä on testitiedosto, joka on luotu git init:n jälkeen.

+Tämä on lisäys commitin jälkeen.

\ No newline at end of file

## Virhetilanteet Gitin kanssa

Mikäli virhetilanteita sattuu Gitin hallinnassa, niitä voidaan korjata eri tavoin.

Mikäli halutaan lisätä muutoksia viimeisimpään commit-komennon ajoon, voidaan käyttää optiota --amend. Tämä käynnistää editorin commitille, lisää muutokset ja ajaa uudestaan commitin.

$ git commit --amend

[master 23c9eb8] Lisätty ominaisuus testi

Date: Sun Jan 22 16:48:13 2023 +0200

1 file changed, 1 insertion(+)

create mode 100644 testitiedosto.txt

$ git diff testitiedosto.txt

diff --git a/testitiedosto.txt b/testitiedosto.txt

index 5a192f0..4e0c09b 100644

--- a/testitiedosto.txt

+++ b/testitiedosto.txt

@@ -1 +1,3 @@

-Tämä on testitiedosto, joka on luotu git init:n jälkeen.

\ No newline at end of file

+Tämä on testitiedosto, joka on luotu git init:n jälkeen.

+Tämä on lisäys commitin jälkeen.

+Tämä lisäys tulee amendin jälkeen.

\ No newline at end of file

commit-komento voidaan myös perua esimerkiksi yhden tiedoston osalta. Ennen komennon ajoa tiedosto toinen.txt oli luotu ja lisätty add-komennon avulla.

$ git reset HEAD toinen.txt

Unstaged changes after reset:

M testitiedosto.txt

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: testitiedosto.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

toinen.txt

no changes added to commit (use "git add" and/or "git commit -a")

Jos halutaan palauttaa tila ennen commit-komennon ajoa, se voidaan tehdä komennolla checkout.

$ git checkout -- testitiedosto.txt

$ git status

On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed)

toinen.txt

nothing added to commit but untracked files present (use "git add" to track)

## Gitin haarautuminen (branch)

Gitissä on käytössä version haarauttaminen, jolloin alkuperäisestä versiosta voidaan jakaa eri haaroja, joissa työskennellään saman aikaisesti ja jopa eri kehittäjien kanssa samaan aikaan.

Haaran voi luoda branch-komennon avulla.

$ git branch haara1

Eri haarojen välillä voidaan siirtyä checkout-komennolla.

$ git branch

haara1

\* master

$ git checkout haara1

Switched to branch 'haara1'

Vaihtoehtoisesti haaran voi luoda ja samalla siirtyä uuteen haaraan.

$ git checkout -b haara1

Lisätään uuteen haaraan tiedosto ja tehdään sille commit.

$ git add haara1.txt

$ git commit -m "haara1.txt lisätty"

[haara1 4b42e81] haara1.txt lisätty

1 file changed, 1 insertion(+)

create mode 100644 haara1.txt

Eriytyneet haarat voi nähdä seuraavan komennon avulla:

$ git log --graph --all --oneline

\* 4b42e81 (HEAD -> haara1) haara1.txt lisätty

\* 23c9eb8 (master) Lisätty ominaisuus testi

Haarat on mahdollista yhdistää merge-komennon avulla.

$ git checkout master

Switched to branch 'master'

$ git merge haara1

Updating 23c9eb8..4b42e81

Fast-forward

haara1.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 haara1.txt

Näin muutokset yhdistettiin automaattisesti. Konfliktitapauksissa on pakko tehdä yhdistäminen manuaalisesti editorin avulla. Kun konfliktit ovat ratkaistu, normaali commit-komento yhdistää haarat.

Haaroja voidaan seurata branch-komennon avulla.

$ git branch -v

haara1 4b42e81 haara1.txt lisätty

haara\_yhdistamaton 1f9f70b yhdistamaton tiedosto

\* master 4b42e81 haara1.txt lisätty

$ git branch --merged

haara1

\* master

$ git branch --no-merged

haara\_yhdistamaton

Haaroja voidaan poistaa tarvittaessa.

$ git branch -d haara1

Deleted branch haara1 (was 4b42e81).

$ git branch **-D** haara\_yhdistamaton

Deleted branch haara\_yhdistamaton (was 1f9f70b).

$ git branch -v

\* master 4b42e81 haara1.txt lisätty

Tässä tapauksessa kaikki haarat ovat poistettu versionhallinnasta (seurattu ja seuraamaton).

## Etä-repository (remote)

Jos etä-repository on pilvessä, sen voi kopioida paikalliselle repositorylle kloonaamalla.

$ git clone https://github.com/leksu70/2023OhjProj1-testi.git

Cloning into '2023OhjProj1-testi'...

remote: Enumerating objects: 3, done.

remote: Counting objects: 100% (3/3), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Receiving objects: 100% (3/3), done.

Tässä syntyneen hakemiston sisältö esimerkkinä.

$ cd 2023OhjProj1-testi

$ ls -Flga

total 9

drwxr-xr-x 1 197610 0 Jan 22 17:53 ./

drwxr-xr-x 1 197610 0 Jan 22 17:53 ../

drwxr-xr-x 1 197610 0 Jan 22 17:53 .git/

-rw-r--r-- 1 197610 47 Jan 22 17:53 README.md

Käytettävän etä-repositoryn tiedot näkyvät remote-komennolla.

$ git remote -v

origin https://github.com/leksu70/2023OhjProj1-testi.git (fetch)

origin https://github.com/leksu70/2023OhjProj1-testi.git (push)

Mikäli haluat itse siirtää paikallisesta repositorysta pilvipalveluun etä-repositoryn, se onnistuu muutamalla komennolla git-hallinnoimasta hakemistosta (repositorysta) käsin. Tämä repository täytyy olla luotuna ensin Gitissä.

$ git remote add origin https://github.com/leksu70/2023OhjProj1-testi-remote.git

$ git remote -v

origin https://github.com/leksu70/2023OhjProj1-testi-remote.git (fetch)

origin https://github.com/leksu70/2023OhjProj1-testi-remote.git (push)

$ git push -u origin master

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Writing objects: 100% (3/3), 270 bytes | 270.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/leksu70/2023OhjProj1-testi-remote.git

\* [new branch] master -> master

branch 'master' set up to track 'origin/master'.

Kaikki haarat voidaan synkronoida etä-repositoryyn yhdellä komennolla.

$ git push --all

Etä-repository voidaan tarpeen vaatiessa uudelleennimetä (origin 🡪 temppi) tai poistaa.

$ git remote rename origin temppi

Renaming remote references: 100% (1/1), done.

$ git add .

$ git commit -m "remote-test2.txt lisätty"

[master ae181f0] remote-test2.txt lisätty

1 file changed, 1 insertion(+)

create mode 100644 remote-testi2.txt

$ git remote -v

**temppi** https://github.com/leksu70/2023OhjProj1-testi-remote.git (fetch)

**temppi** https://github.com/leksu70/2023OhjProj1-testi-remote.git (push)

$ git push -u temppi master

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 4 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 331 bytes | 331.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/leksu70/2023OhjProj1-testi-remote.git

4806927..ae181f0 master -> master

branch 'master' set up to track 'temppi/master'.

$ git remote rm temppi

$ git remote -v

Mikäli etä-repository on poistettu paikallisesta käytöstä, silloin repository voidaan palauttaa hakemalla se uudelleen.

$ git remote add origin https://github.com/leksu70/2023OhjProj1-testi-remote.git

$ git fetch origin

From https://github.com/leksu70/2023OhjProj1-testi-remote

\* [new branch] master -> origin/master

$ git branch -v

\* master ae181f0 remote-test2.txt lisätty

$ git branch -r

origin/master

## Omia mietintöjä

Tässä dokumentissa on esitelty Gitin toimintaa. Itselleni uutta olivat haaran käsittelyt ja koin tämän dokumentin teon parantavan ymmärrystä versionhallinnasta git-hubin avulla.