Fontys Git Manual



Last version: 19-01-2021

Authors:  
Tim de Laat  
Kristian Snel

CONTENT

[1. Manual structure 3](#_Toc61985435)

[2. GitHub Basic 4](#_Toc61985436)

[2.1. Github.com introduction 4](#_Toc61985437)

[2.2. Create a repository 4](#_Toc61985438)

[2.3. Github projects 5](#_Toc61985439)

[2.4. Github issues 5](#_Toc61985440)

[2.5. Github wiki 5](#_Toc61985441)

[3. Git Technical 6](#_Toc61985442)

[3.1. Installation 6](#_Toc61985443)

[3.2. Version control with Git 7](#_Toc61985444)

[3.3. Cloning an existing repository 8](#_Toc61985445)

[3.4. Git workflow 9](#_Toc61985446)

[3.5. Branching 10](#_Toc61985447)

[3.6. Git Issues 11](#_Toc61985448)

[3.7. Github insights 12](#_Toc61985449)

[4. Git Management 12](#_Toc61985450)

[5. Git Advanced 12](#_Toc61985451)

[6. For Teachers 13](#_Toc61985452)

[6.1. After the project 13](#_Toc61985453)

[6.2. Continuing a project 13](#_Toc61985454)

[6.3. Discontinuing a project 13](#_Toc61985455)

[7. Extra info 13](#_Toc61985456)

[7.1. Issue Templates 13](#_Toc61985457)

[7.2. Release Templates 13](#_Toc61985458)

[7.3. Git commands 14](#_Toc61985459)

[8. Appendix 16](#_Toc61985460)

[8.1. A. Installation 16](#_Toc61985461)

[9. Bibliography 19](#_Toc61985462)

# What is Git and how does it work

Git helps improve your project structure and allows groups to work more efficiently on project files and documents. Git is described as a version control system.

### About Version Control

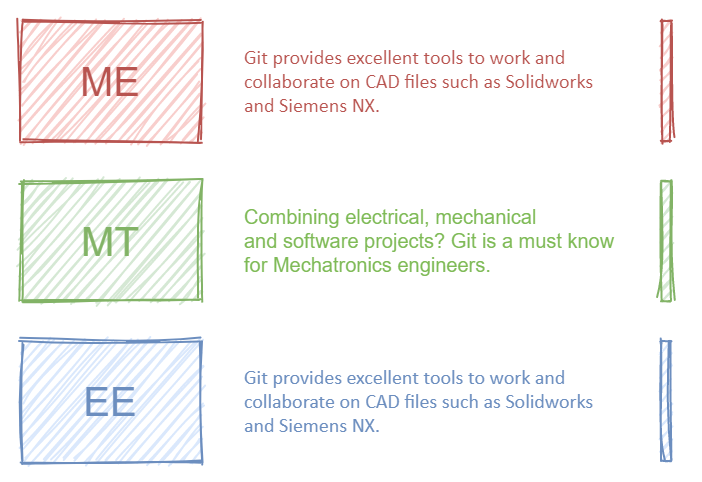
What is “version control”, and why should you care? Version control is a system that records changes to a file or set of files over time so that you can recover files to their historic versions. This helps with implementing new features and still allows you to look back at earlier concepts of your work. This can be helpful in case an addition to a certain module causes the system to fail and changes have to be reverted.

### Git in EXPO projects

To improve the way of working within EXPO groups all disciplines are required to take knowledge of Git. This allows you to work together more efficiently and get started right away without discussing the need for a digital file storage solution.

To familiarize every student with Git first a Git basic introduction is given. This chapter requires you to install Github Desktop and create an account at Github.com. If you have mastered a basic workflow in Git and feel that you want to learn more about Git, this manual also features two chapters going deeper into the Management and Technical uses of Git.

### Department specific tips

Git was originally intended for use with Software but is much more:

Mechanical Engineering

Mechatronics engineering

Electrical engineering

# GitHub Basic

## Github.com introduction

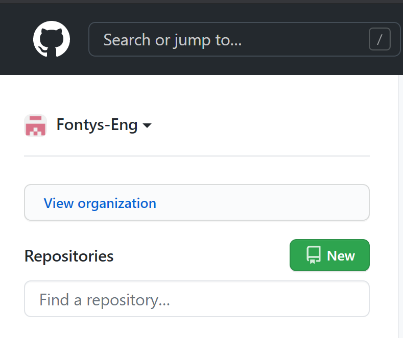
Create **an** account

The first step is to create a GitHub account. Go to <https://github.com/> and press “Sign Up” in the right top corner. Enter your email, username, password and solve the puzzle.

Optional: activate GitHub education pack for free!

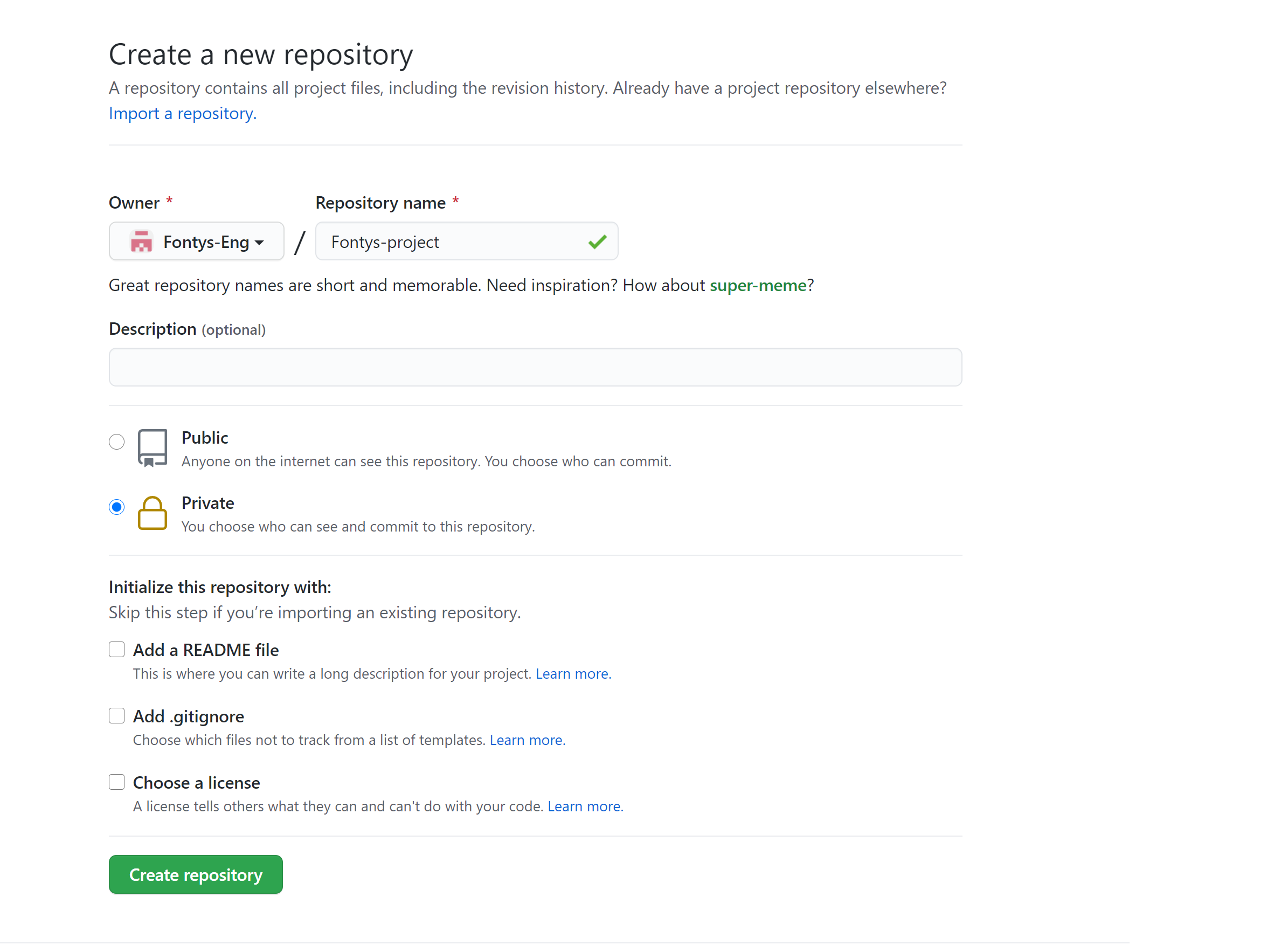
One of the benefits of being a student with GitHub is that can get software tools for free. You can sign-up via <https://education.github.com/pack>. One of the primary perks is GitHub pro, which allows you to add more people to a private repository (free GitHub only allows up to three collaborators).

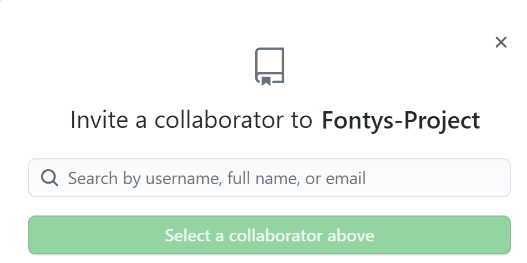
## Create a repository

Manage repository structure

The next step is to create a repository. A repository is a version controlled project folder, but this is used for now. GitHub offers a lot of project management tools with a repository. Things like projects boards, issue pages and wiki’s.

To create a repository, login with your GitHub account and press new in the left top of your screen. See the image below. Give the repository a name, add a description and click on repository “Private”. With a Private repository, the owner can decide who can see and work in the repository. Only one person of each EXPO-group should make a repository. This repository is handed to your teacher at the end of the project.



Manage access

The next step is to invite your other group-members, tutor and teacher into the repository. To do that, open your repository. Go to settings🡪Manage access and press Invite a collaborator. You should see a similar prompt to this image.

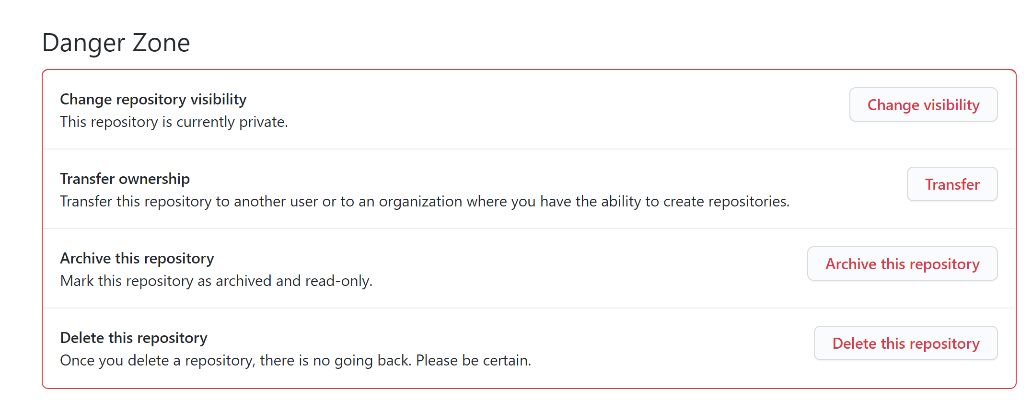
Choose to invite the new group into the repository (and remove the old group from it). possible for the teacher to hand-over the repository to the new group, or let them make a new repository.

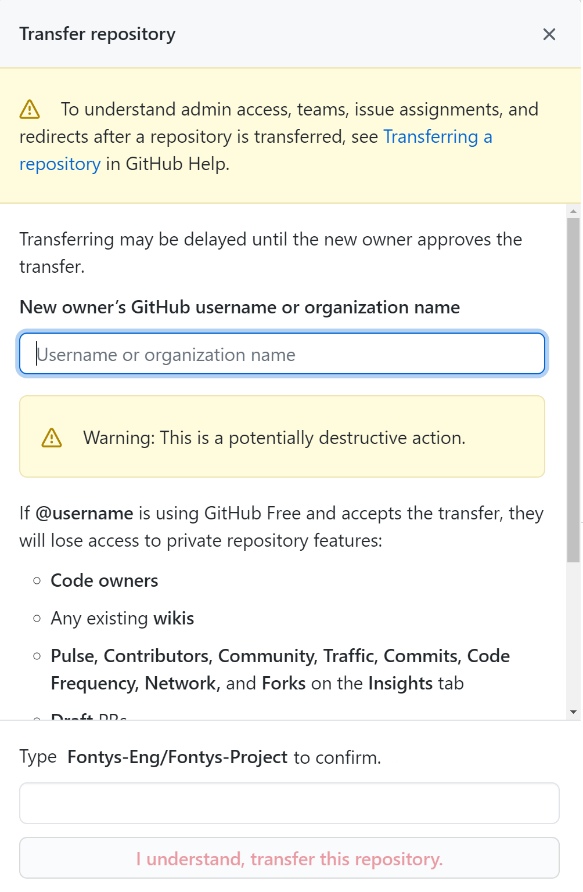
## Github projects

## Github issues

## Github wiki

Transfer ownership

For continuation of the project, it is important to hand-over the repository to your teacher. First you have to make sure that your teacher has a GitHub pro account. This is due to collaborator limit in private repositories with free GitHub accounts. To check this, go to Manage access and hove over the profile picture. If it is a pro user, it should show the  icon.

To transfer ownership, the owner of the repository must go to settings, and scroll all the way down to the “Danger Zone” and press “Transfer”. A prompt should show up for transferring the repository. There, enter the GitHub username of your teacher. Always double check if you have the user selected. Then follow the other instruction and press “I understand, transfer this repository.” After this is done, inform your teacher that you made a request to transfer your repository. He/she should have received an email about this to accept the transfer.

# Git Technical

## Installation

Git supports a great variety of Desktop clients. One of the most popular interfaces is Github Desktop.

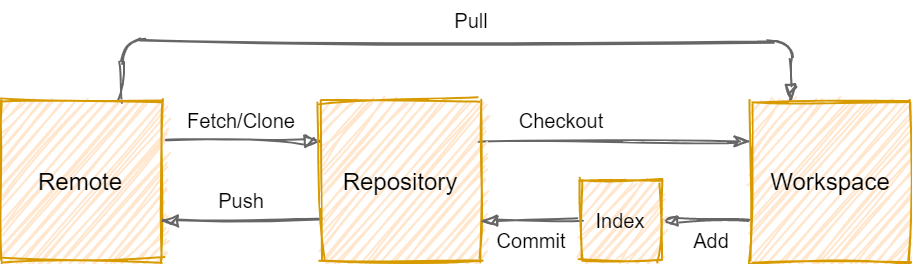
Github Desktop can be downloaded using this link: <https://desktop.github.com/>

For full details on how to install Github Desktop please refer to Appendix A. Installation.

After downloading the software create a GitHub account at <https://github.com/join?source=login>

## Version control with Git

Before using Git it is important to understand a few terms and concepts.



**Repository** A repository (or repo) is a folder or “storage location” with a .git/ folder inside to add Git version-control. A repository records versions of the files that it contains and may support multiple timelines of a project (see Branches).

**Local**

**Origin**

**Remote** A remote is a project repository that is not on your computer.

**Upstream**

**Commit** A commit used to save your changes to the local repository. To commit changes, you select one or more changed files and give those changes a descriptive title/summary. Optionally you can add a description to your changes. To upload local changes, see git Push.

**Push** Pushes your commits to your origin repository.

**Fetch** Downloads the commits, files and refs from a remote repository into your local repo.

**Pull** Downloads the commits, files and refs from a remote repository into your local repo and merges this with your local branches.

**Clone** Clones a repository into a newly created directory, creates remote-tracking branches for each branch in the cloned repository (visible using git branch --remotes), and creates and checks out an initial branch that is forked from the cloned repository’s currently active branch.

**Pull Requests** A Pull Request (PR) is a request to ask your upstream project to pull your changes into their tree. Usually, a developer makes a pull-request when they finished a feature, or made a hotfix. The developer should add summary On GitHub, you can select one or more persons to review your changes and accept your changes.

**Stash** If two people edit the same file at the same moment a file conflict may occur. Unfortunately, this may still always occur. In this case those conflicts will be listed under the Stashed changes tab.

Now you are ready to install Github Desktop and start your first Git project.

Graphical user interface, text

Description automatically generated

Setting up a test project

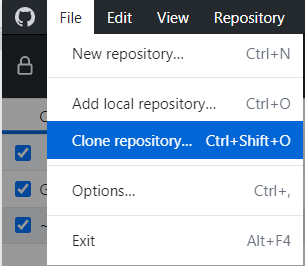


## Cloning an existing repository

To continue working on a project but create a copy of your own you may consider cloning an existing repository. To clone an existing repository, press the following button after installing the software.



Or this button under File 🡪 Clone repository ( + O)



To clone a repository, find a repository online, enter a url or select one of your own repositories.

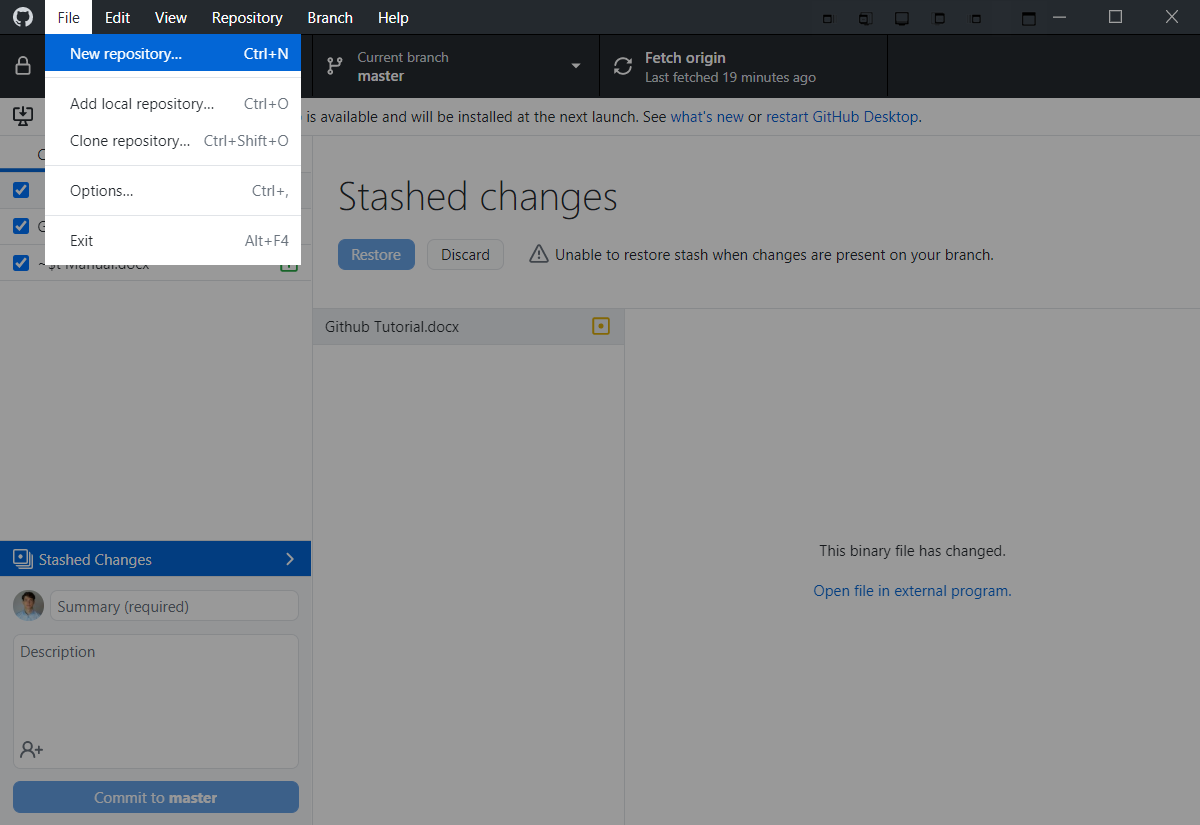
Then simply press the clone button and watch as the clone is created and the repository is opened.

Setting up a new repository

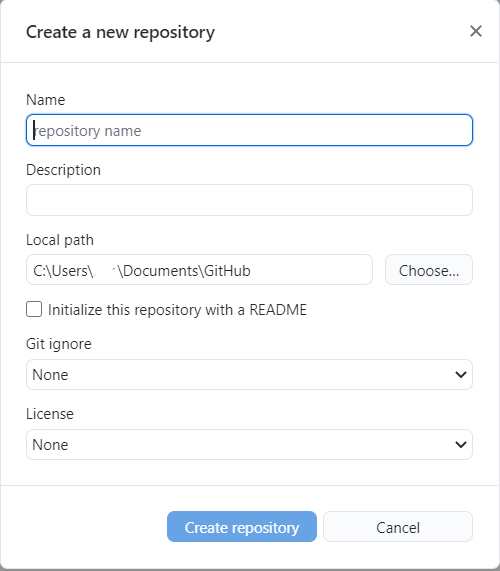
To create a new Repository right after installing Github Desktop click the button shown in the picture below.



If you have already opened a project, create a new repository under File 🡪 New repository



A new repository needs a Name and a description. By default, the new Repository will be saved locally in the Users\#YourPC#\Documents\GitHub folder.



## Git workflow

Push pull requests

Git functions by users sending updates and receiving them periodically. The power of Git is in the ability to update and receive files when you decide so. When using Git, an update from others is usually called a Pull request. While sending changed files over to the server or other people is generally called a Push request. If a user wishes to see if any files changed over at their team members, they could perform a so-called Fetch request. Therefore, A Pull request is always followed up by a Fetch request.

A Push request is initiated by committing. Committing is the process of submitting your changed files with a description about what changed.

## Branching

Master branch

A branch is best explained as a separate timeline of a project. To allow multiple persons to work on the same project different branches may be desirable to prevent conflicts and allow testing of individual functionality. A user may create a branch, add changes, test those and then suggest the updated content to be used in the master branch. The master branch is where usually all branches come together and form the final project files. Updating a separate branch into the master branch is called merging.

Diagram

Description automatically generated

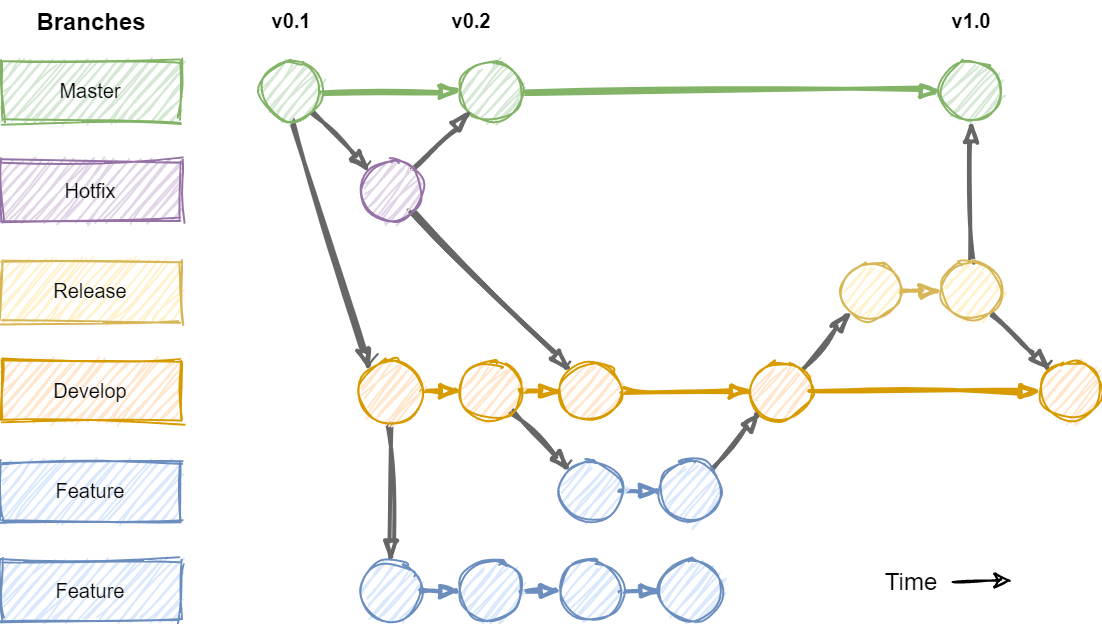
**Branch** A branch is an alternative timeline of a Repository (or project folder). This means that you may work on the same project but can also view the old unchanged files as they were before “branching”.

**Merge** …

**Rebase**  …

**Merge conflict …**

## Git Issues



## Github insights

# Git Management

Semantic versioning

* **Major, Minor, Patch**

Tags and Releases

* **Releases**
* **Closing Issues**
* **Compatibility Matrix**

Readme.md

* **Markdown**

Project Boards

* **Project board template**
* **Link Issues**

Milestones

* **Project board template**

Insight

* **GitHub insights**

# Git Advanced

.gitignore

Git Submodules

Git Bash

# For Teachers

This chapter is meant for teachers. It give tips on how to manage the repository that is received from the EXPO group, and how to manage it when a project continues or discontinues.

## After the project

When the teacher receives a repository, it is recommended to remove the group members from the previous EXPO group before adding the new members. If the project member still wants to keep a copy of the project, it can choose to clone by making a Fork. This will only copy the files, not the issues, project boards and wiki documentation. For it is recommended to give the students a heads-up before they are kicked out.

## Continuing a project

When a project continues, there are three options to deal with the

* The teacher invites the new EXPO group (and new Tutor) to the repository and maintain ownership (Recommended). The EXPO group will either create a new project board, or continue using the project board of the previous group. It will has access to see both open and closed issues, their project board. This shows the progress of the previous group, as well as their achievements, struggles and design discussions (trough closed issues). The group also has access to the and possible wiki documentation.
* The teacher hands over the repository to the students. This has the same benefits as discussed above, but the project is in control of the repository. This can add the risk of losing the repository. Before doing this, make sure that the student that receives the repository has GitHub pro (if not, recommend him to take activate the GitHub activation pack).
* The last option is to ask the new EXPO group to create a new repository. The new group starts with a clean slate, but has no insight in the activities from the previous group (besides what is mentioned in the report).

## Discontinuing a project

When a project is discontinued, it is recommended to Archive the repository. This will make the repository read-only, preventing changes to be made to it afterwards. It is recommended by GitHub to close all active pull-requests and open issues first. To archive a report, go to settings, scroll down to the “Danger Zone”, follow the instructions and press “I understand the consequences, archive this repository”. It is possible to unarchive a repository afterwards.

# Extra info

## Issue Templates

## Release Templates

|  |
| --- |
| \_Software v2.13.0 release\_  # What's New **✨**  - Add Hello World (#152 )  - Fix old stuff (#123)  # Enhancements **🙌**  ## Add new stuff (#152 )  Added a function to print Hello world in the terminal.  Closes #152  # Bugs/issues Fixes 🐛  ## Fix old stuff (#123)  Fixed issue where the stops working after user entered a character.  Closes #123  # Features to look into 🔎  ## In progress  - Refactor Hello World(On Hold).  - Update Comments  ## Known issues  - Something is wrong. |

## Git commands

git init

git clone git pull

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

git touch documentname

git add wildcard \*.html all .

git status

git commit -m 'description what changed' (Waar is die -m ook al weer voor?)

git push

Branching

git branch branchname

git checkin git checkout

gitmerge branchname 'describe merge'

# Appendix

## A. Installation



Run the downloaded GitHubDesktopSetup.exe

Sign in to or create a free account.





Log in to your account.



# Bibliography

|  |  |
| --- | --- |
| [1] | Backlog, "Git vs. SVN: Which version control system is right for you?," 23 June 2020. [Online]. Available: https://backlog.com/blog/git-vs-svn-version-control-system/. |
| [2] | Git, "Git - About Version Control," Git, [Online]. Available: https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control. |
| [3] | Atlassian, "Git Merge," Atlassian, [Online]. Available: https://www.atlassian.com/git/tutorials/using-branches/git-merge#:~:text=Merging%20is%20Git's%20way%20of,them%20into%20a%20single%20branch.&text=The%20current%20branch%20will%20be,branch%20will%20be%20completely%20unaffected.. |