

GitLab Introduction

Jäkel, Anna Christina

April 2020

Contents

1	Introduction	2
2	Git	2
3	Install Git	3
4	Clone a Repository to your Computer	3
5	Using a Git Repository	5

1 Introduction

GitLab is a web application based on Git, which is a distributed version control system for tracking changes in source code. It is used by the LRZ, therefore you are automatically registered with your TUM account and can start with it right away.

GitLab can be used as a tool for collaborative software management as well as a version control system. The Terms of Service for the LRZ-hosted GitLab can be found here: https://www.lrz.de/services/netzdienste/gitlab_en/. To use the LRZs GitLab follow this link https://gitlab.lrz.de/users/sign_in and sign in with your TUM account. After you got access to a repository of the owner you should see all the projects on the startpage. An example is shown in figure 1.

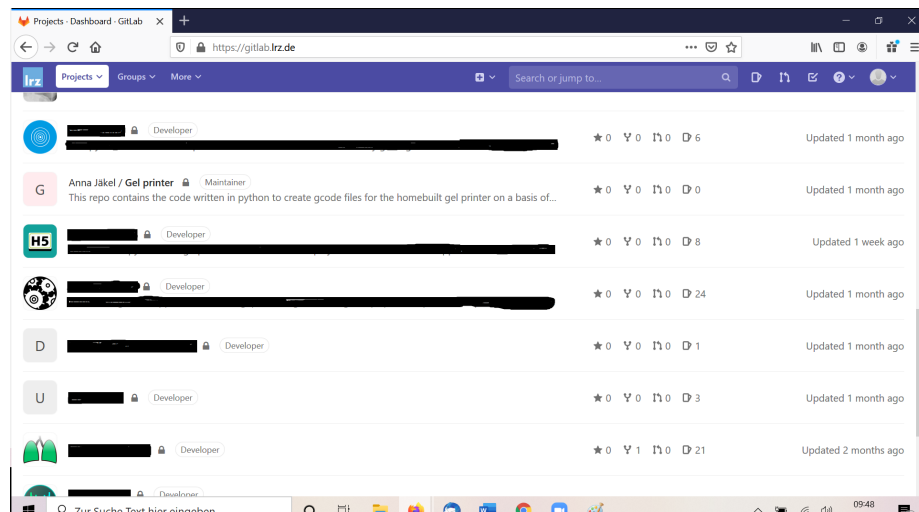


Figure 1: Startpage of the lrz gitlab after you signed in and got access to various projects.

2 Git

Git is the most widely used modern version control system. It is an open source project developed in 2005 by Linus Torvalds. Git has many interesting features. The most famous one is the Branching and Merging. What is meant by Branching? Branching means that you can have multiple local branches that can be entirely independent of each other. A branch is a copy of your whole code. It is very usefull when you want to try out something different with your code, but want to keep your original code in parallel so that you can still use it. After you successfully tried out something on your new branch you can merge those two, which means that you add your changes to your original code.

3 Install Git

Usually Git is already installed on Mac OS and you don't need to install it. For windows you need to install it. Browse to the official Git website <https://git-scm.com/downloads> and click on the download link for windows and allow the download to complete. Double click the installation file, which you usually find in you download folder. Allow the app to make changes to your device and follow the instructions of the installer. A more detailed instruction can be viewed on this website for exampple <https://phoenixnap.com/kb/how-to-install-git-windows>.

4 Clone a Repository to your Computer

To view one of the projects just click on it and the repository will open, it will look like figure 2. If you want to use the code in the repository on your

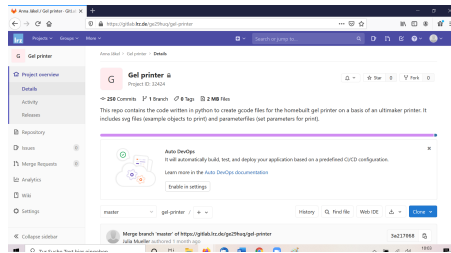


Figure 2: Startpage of a repository.

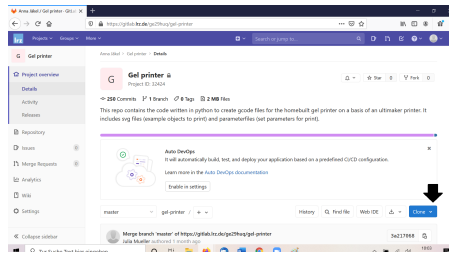


Figure 3: Clone a repository.

computer, you can clone the repository to your computer as follows. Click on the clone button on the right, as shown in figure 3. A small window will open which shows Clone with SSH or HTTPS. For simplicity we will use "Clone with HTTPS". Copy the link shown above that option - this is the url to the repository.

Now you have to open a command line on you computer. For windows use the key combination windows + R and type cmd in the window (For Mac OS you can type "Terminal" in the search function). A command line will open, this will look like figure 4. With the command `git --version` you can check if git is available in this command line (if you followed the instructions of the website <https://phoenixnap.com/kb/how-to-install-git-windows> that should work). If not you have to add git to your PATH variable. As an alternative you should be able to open a command line via right clicking on a window and selecting *git bash here*. For our application it doesn't matter with command line you will use, just make sure that your are in the correct folder.

With the command `cd path_of_the_folder_where_the_repository_should_be_cloned_to` you swap to that specific location. Now you can clone your repository with the command `git clone url_that_you_copied_in_the_step_before`. The commands are shown in figure 5. After clicking the enter button

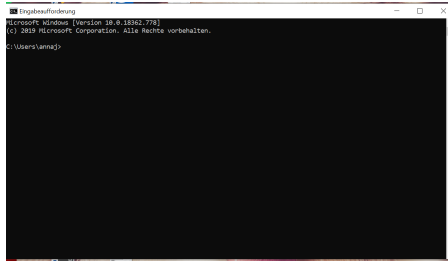


Figure 4: Command line.

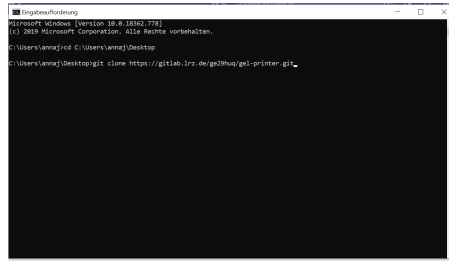


Figure 5: Commands to change the directory and to clone a repository.

a window will pop up or you will be asked in the command line to enter your LRZ-username and password (see figure 6).

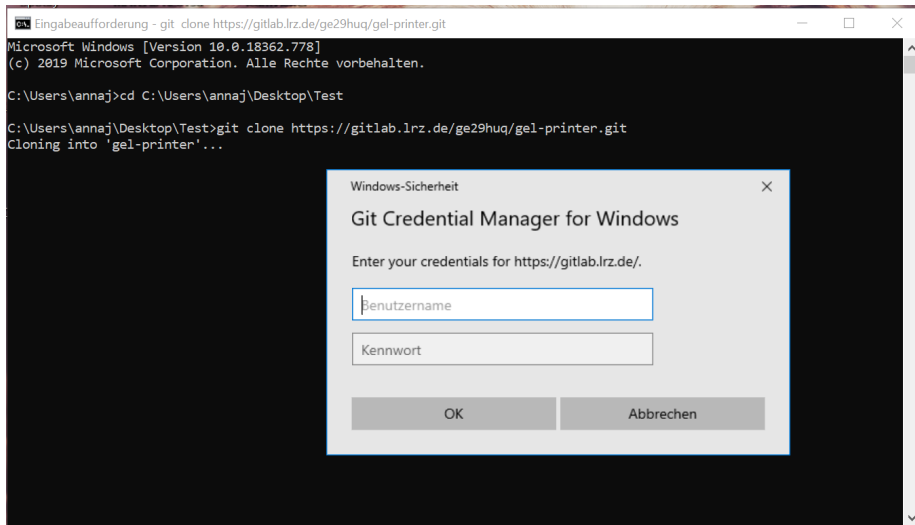


Figure 6: To successfully clone a repository you will be asked to authenticate yourself. After you got access to the specific repository by the owner you can use your LRZ-credentials here.

5 Using a Git Repository

After you successfully cloned the whole code of the repository, you can use it as you used code before. If you want to make contributions to the code and upload or change something you can do that with the command line. In the following the most important commands are presented (In figure 7 and following a whole working-flow example is shown):

- **git status** : checks the status of your cloned repository, it tells you if your repository is up to date or if you have to make commits
- **git pull** : pulls the repository from its original locations, that means it downloads additional changes that were made to the repository
- **git add filename** : adds a file to the repository
- **git add .** : all files that are not in the file control yet will be added
- **git rm filename** : removes a file from your repository
- **git commit -m "commit message"** : commits your changes so that those can be pushed to the original repository with a committing message so that other users know what changes you made
- **git push** : pushes your changes to the repository to the original one

If you want to use further functions of git a good documentation is given by Scott Chacon and Ben Straub. Their book is available on GitHub <https://git-scm.com/book/en/v2>.

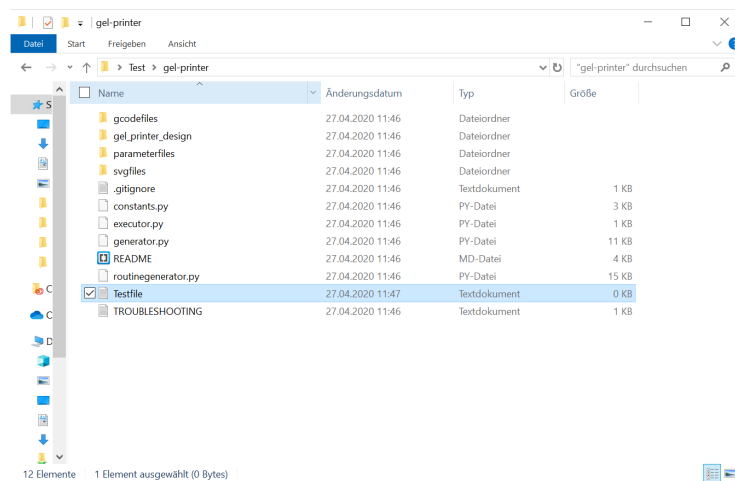
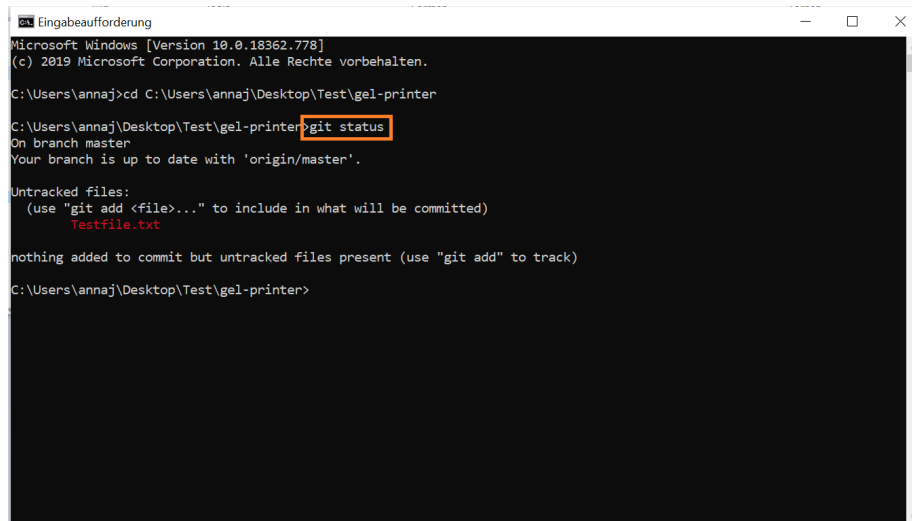


Figure 7: In this example a simple textfile is created with the name *Testfile*.



```
Eingabeaufforderung
Microsoft Windows [Version 10.0.18362.778]
(c) 2019 Microsoft Corporation. Alle Rechte vorbehalten.

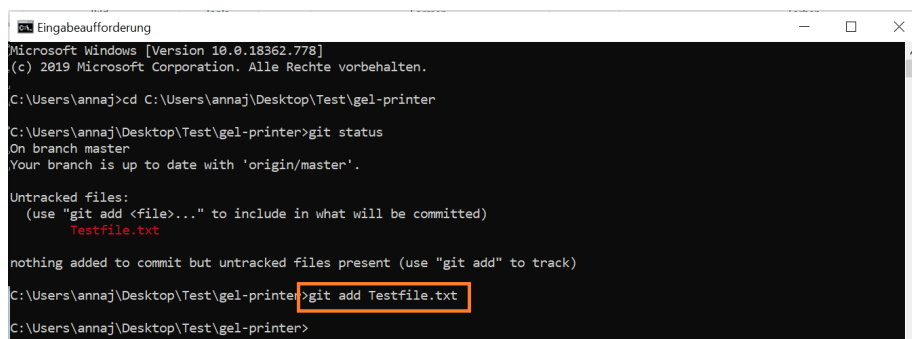
C:\Users\annaj>cd C:\Users\annaj\Desktop\Test\gel-printer

C:\Users\annaj\Desktop\Test\gel-printer>git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Testfile.txt

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 8: After adding this file you can check with *git status* what is the status of your repository. You have added the file *Testfile.txt* but it is not added to git's version control yet.



```
Eingabeaufforderung
Microsoft Windows [Version 10.0.18362.778]
(c) 2019 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\annaj>cd C:\Users\annaj\Desktop\Test\gel-printer

C:\Users\annaj\Desktop\Test\gel-printer>git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Testfile.txt

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\annaj\Desktop\Test\gel-printer>git add Testfile.txt
C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 9: If you want to add this file to your git repository type *git add Testfile.txt* in the command line.

```
Eingabeaufforderung
Microsoft Windows [Version 10.0.18362.778]
(c) 2019 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\annaj>cd C:\Users\annaj\Desktop\Test\gel-printer

C:\Users\annaj\Desktop\Test\gel-printer>git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Testfile.txt

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

C:\Users\annaj\Desktop\Test\gel-printer>git add Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 10: With the command *git status* you can check now how the status of repository has changed after adding the file. The testfile is added to gits version control now, but the changes aren't committed yet. This has to be done, so that it can be pushed to the original repository.

```
Eingabeaufforderung

C:\Users\annaj\Desktop\Test\gel-printer>git add Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git commit -m "New Testfile added."

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'annaj@DESKTOP-BB75F8T.(none)')

C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 11: With the command *git commit -m "Your commit message should be typed in here"* you can commit all you changes that you made and added, in this example the addition of the festfile. When committing the first time an error will rise, because you have to authenticate yourself again. In the next step it is shown how you can deal with that.

```
Eingabeaufforderung
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git commit -m "New Testfile added."

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'annaj@DESKTOP-BB75F8T.(none)')

C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.email anna.jaekel@tum.de
C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.name ge29huq
C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 12: You can set your email address and username globally for this repository. Just use the commands shown in the error message: *git config --global user.email "you@example.com"* and *git config --global user.name "Your Name"*. This should be your TUM email address and your TUM credentials. After setting your email address and username you won't be asked again to authenticate when committing changes.

```
Eingabeaufforderung

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git commit -m "New Testfile added."

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

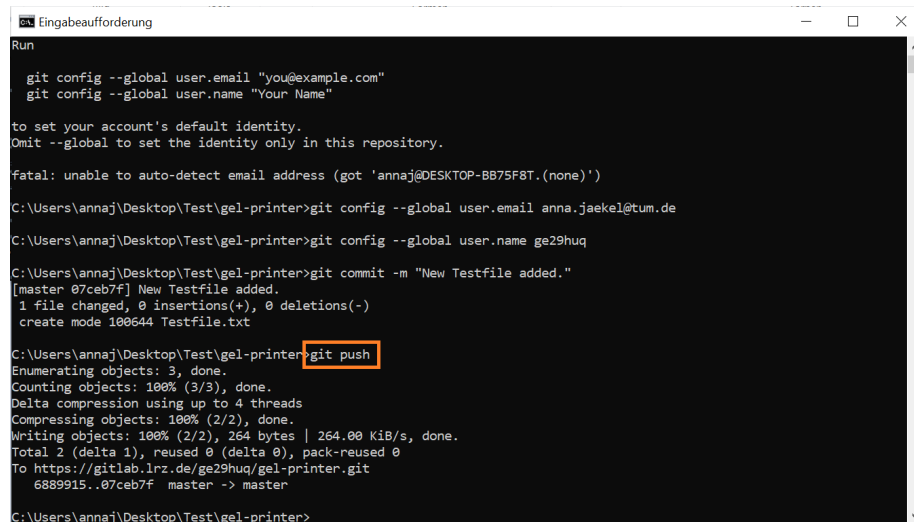
to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'annaj@DESKTOP-BB75F8T.(none)')

C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.email anna.jaekel@tum.de
C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.name ge29huq
C:\Users\annaj\Desktop\Test\gel-printer>git commit -m "New Testfile added."
[master 07ceb7f] New Testfile added.
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 13: After setting your email address and username globally you can commit with the command mentioned before. This time a positive answer of git should be shown as in this example.



```
Run
git config --global user.email "you@example.com"
git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

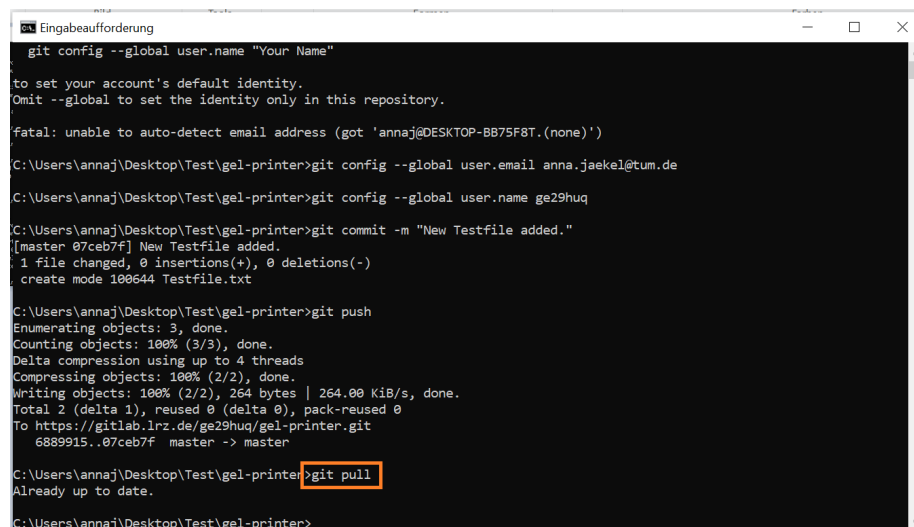
fatal: unable to auto-detect email address (got 'annaj@DESKTOP-BB75F8T.(none)')

C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.email anna.jaekel@tum.de
C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.name ge29huq
C:\Users\annaj\Desktop\Test\gel-printer>git commit -m "New Testfile added."
[master 07ceb7f] New Testfile added.
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 264 bytes | 264.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
To https://gitlab.lrz.de/ge29huq/gel-printer.git
 6889915..07ceb7f master -> master

C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 14: After you committed your changes with a commit message you can push your commit to the original repository with the command *git push*.



```
git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'annaj@DESKTOP-BB75F8T.(none)')

C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.email anna.jaekel@tum.de
C:\Users\annaj\Desktop\Test\gel-printer>git config --global user.name ge29huq
C:\Users\annaj\Desktop\Test\gel-printer>git commit -m "New Testfile added."
[master 07ceb7f] New Testfile added.
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Testfile.txt

C:\Users\annaj\Desktop\Test\gel-printer>git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 264 bytes | 264.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
To https://gitlab.lrz.de/ge29huq/gel-printer.git
 6889915..07ceb7f master -> master

C:\Users\annaj\Desktop\Test\gel-printer>git pull
Already up to date.

C:\Users\annaj\Desktop\Test\gel-printer>
```

Figure 15: Everytime you start working with the gitlab code you should pull the repository before you make any changes, so that the merging of your local repository and the original repository works fine without error messages. Also it can be worthwhile for you as you pull all the changes made by other users that can be helpfull for your work.

References

- [1] Heise. *GitLab vs. GitHub*, 2019 (accessed April 24, 2020).
- [2] Vladimir Kaplarevic. *How to install Git on Windows*, 2020 (accessed April 24, 2020).
- [3] Leibniz Rechenzentrum. *GitLab*, 2018 (accessed April 24, 2020).
- [4] Leibniz Rechenzentrum. *GitLab*, 2020 (accessed April 24, 2020).
- [5] Ben Straub Scott Chacon. *Pro Git*, 2014 (accessed April 24, 2020).