

# gesis

Leibniz Institute  
for the Social Sciences



## Tools and Workflows for Reproducible Research in the Quantitative Social Sciences

### Using Git and GitHub

*Bernd Weiß*

*2022-11-18*

# Overview

# Recap

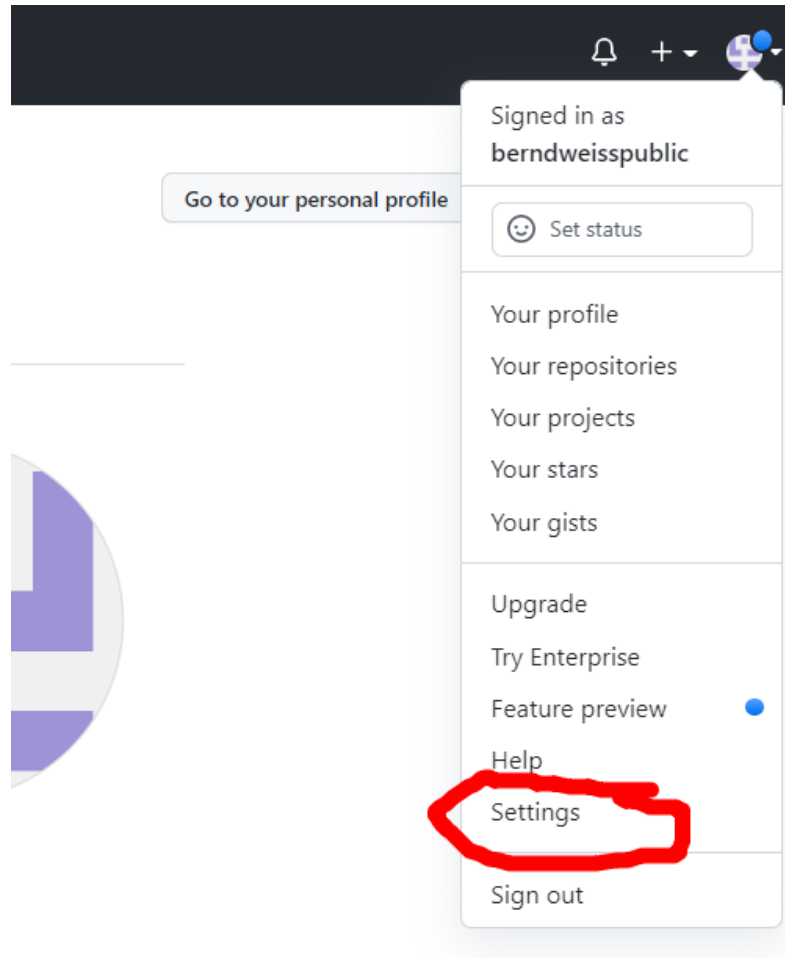
- Basic introduction to Git
- Git workflow
- Important Git concepts/commands (`git init`, `git add`, `git commit`, `git log`, `git show`, `git clone`, `git checkout`, `git branch`, ...)
- ...

# Overview

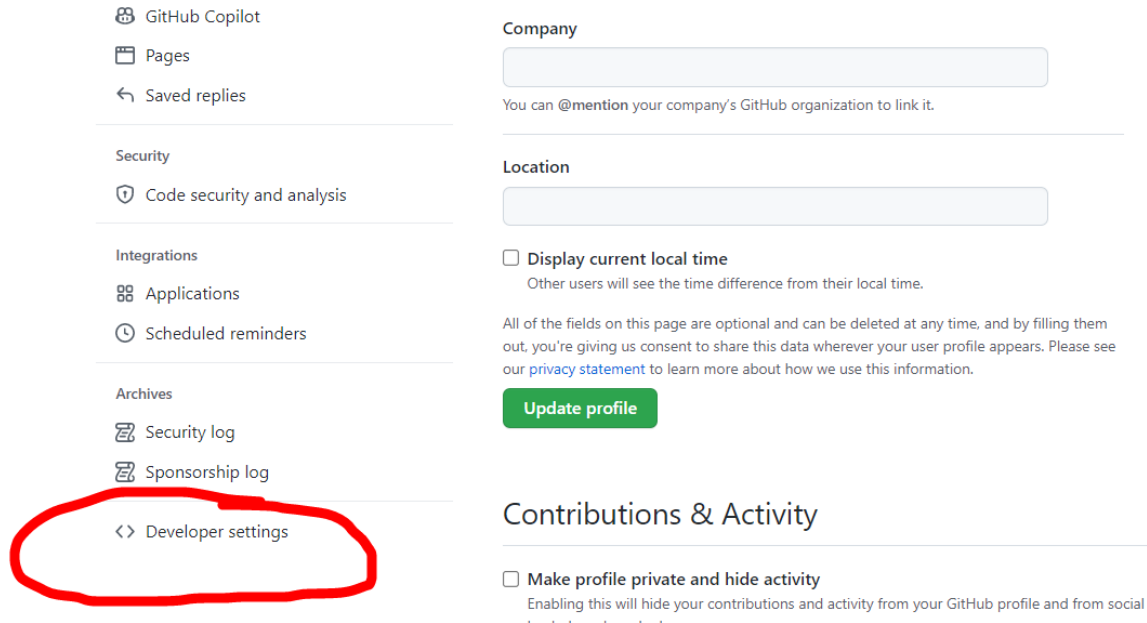
# GitHub: Using personal access tokens

- These days, authentication via personal access tokens (PAT) (and [https](#)) seems the way to go when using GitHub
- In the following, I will illustrate the process using multiple screenshots
- Note that my explanation does not include any R/RStudio-related processes. Johannes will talk about these things in more detail

In GitHub, go to the Settings website:



Next, go to the Developer Settings entry:



The screenshot shows the GitHub Developer Settings page. On the left sidebar, the 'Developer settings' option is highlighted with a red circle. The main content area shows the 'Company' and 'Location' fields, both of which are empty. Below these fields, there is a checkbox for 'Display current local time' which is unchecked. A green 'Update profile' button is visible. The 'Contributions & Activity' section is partially visible at the bottom, showing a checkbox for 'Make profile private and hide activity' which is also unchecked.

**Developer settings**

GitHub Copilot

Pages

Saved replies

Security

Code security and analysis

Integrations

Applications

Scheduled reminders

Archives

Security log

Sponsorship log

**Company**

You can @mention your company's GitHub organization to link it.

**Location**

☐ **Display current local time**  
Other users will see the time difference from their local time.

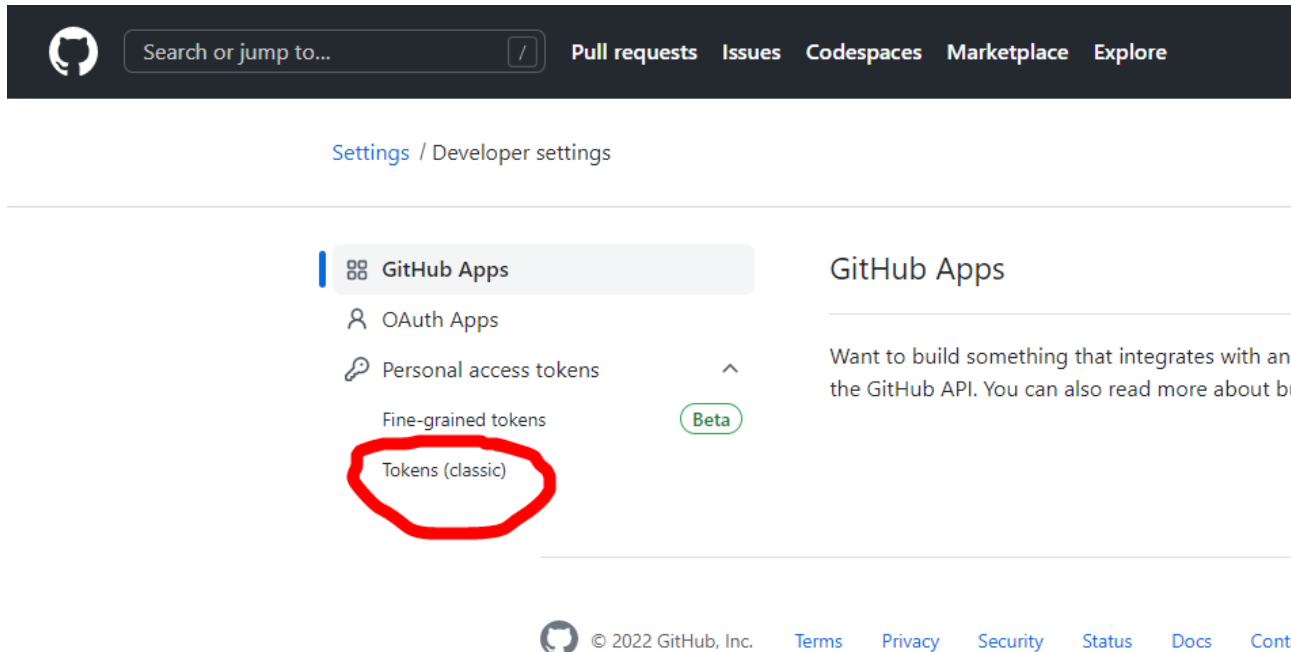
All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our [privacy statement](#) to learn more about how we use this information.

**Update profile**

**Contributions & Activity**

☐ **Make profile private and hide activity**  
Enabling this will hide your contributions and activity from your GitHub profile and from social

Then, choose Tokens (classic):



The screenshot shows the GitHub web interface. At the top is a dark navigation bar with the GitHub logo, a search bar, and links for Pull requests, Issues, Codespaces, Marketplace, and Explore. Below this is a breadcrumb trail: Settings / Developer settings. The main content area is divided into two columns. The left column contains a sidebar with the following items: GitHub Apps (selected), OAuth Apps, Personal access tokens, Fine-grained tokens, and Tokens (classic). The 'Tokens (classic)' item is circled in red. To the right of the sidebar, the 'GitHub Apps' section is visible, with a description: 'Want to build something that integrates with and the GitHub API. You can also read more about bu'. At the bottom of the page is a footer with the GitHub logo, copyright notice '© 2022 GitHub, Inc.', and links for Terms, Privacy, Security, Status, Docs, and Cont...



And, generate a new token; important, save this Token (e.g.,  
ghp\_FqamoGsx6PcEnpwnXyh . . .):

### Personal access tokens (classic)

Generate new token ▼

Need an API token for scripts or testing? [Generate a personal access token](#) for

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used to [authenticate to the API over Basic Authentication](#).

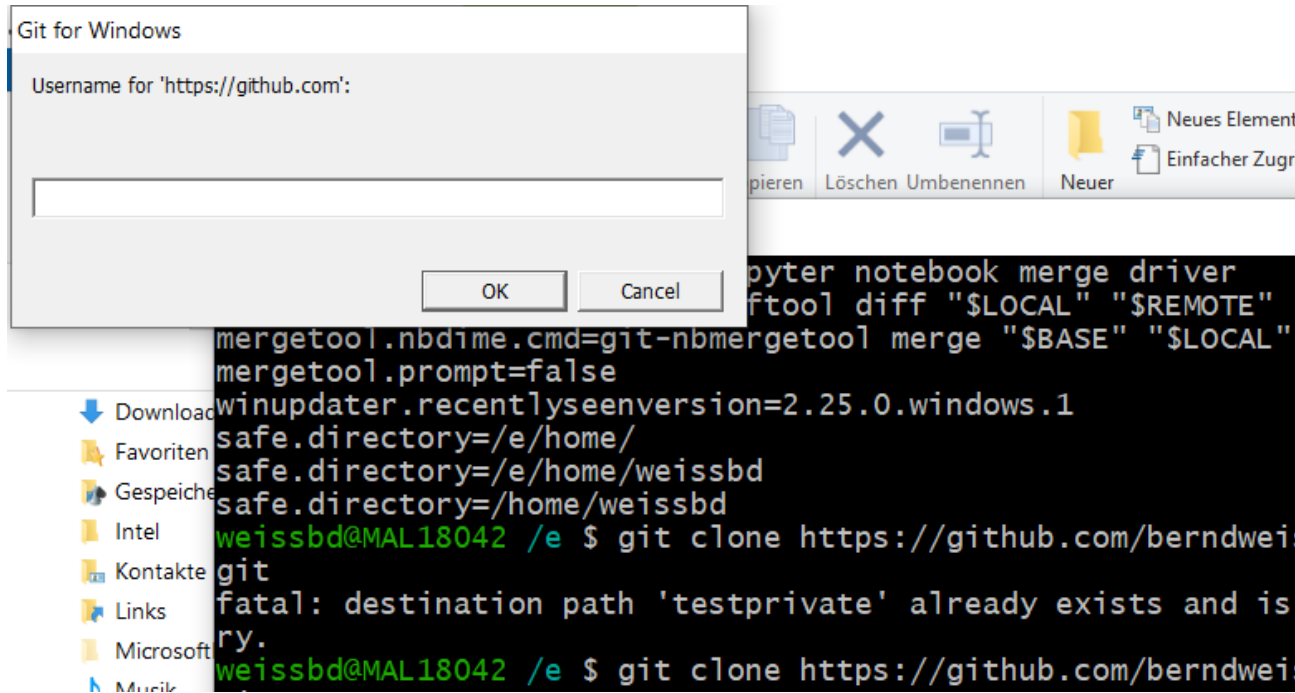
Generate new token Beta

Fine-grained, repo-scoped

Generate new token (classic)

For general use

When you are now cloning a new repository (or pushing/pulling for the first time), you will be asked once to enter your username...



... and your Token (even though it says "Password"):



If, for whatever reason, you decide to reset/remove your credentials, you can do so using the **Windows Credentials Manager** (in German: "Anmeldeinformationsverwaltung")

# Setting up SSH

- SSH is a network protocol that comes in handy, when you work with remote repositories and when you do not want to type-in your password every time you pull (fetch) or push (send) from a remote repository. You still need to authenticate yourself, though.
- To work with Git on you local computer, you do not need SSH (= Secure Shell).
- Authentication in SSH (which is also the name of the program) works by using a private and a public key (usually the public key has the file extension `.pub`, e.g., my public key is `id_rsa.pub`). When you start working with SSH for the very first time, you have to create both keys.

- The private key remains on your local computer and you have to make sure that it is safe -- it is a simple text file and it is your password now, and everyone who has your private key can access your files. Again, everyone who has your private key has your password!

This is what my *public key* looks like:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAY0Q9RT6Tkfgkd02NspzdVJE5CZ03yYAhVwLGo
CrI3E9/Ix0MAySunXExjhsQi2XkhPBjLOEahYuuLaAWHuBc7apUPRNSBy+mdUHnH3
0BdTQijQ6vj3RL99H04yrZnipILkS5ufw/+hpbXX0zS0qTvyGtL9ygm3eA2HDSQtz
2ptFq8an0DDJKrgTbNLb/YZ9KDIcpd0/Sfk4LtvaGF3tIFlyE+pogNmN4eWiYg9Xv
25BhVVxWMHadRFLedastW04SedriEHZQYaNgxVNTufqolJ0nbg4R//fVDxjR2SbzV
AHLZ+eVPUx+vzcPVMP9wYPcni9YLisRy+hLUAOR/kXeQ== berndweiss
```

The *public key* (not the private key!) has to be stored at the GitHub/GitLab/... website. Now, everyone who has your public key can encrypt files (that are sent to you via the internet) but only you (or anyone else who has your private key) can decrypt the files. And, for that reasons you do not have to login everytime you push/pull files from the remote repository.

- How to setup SSH on you computer is explained on this website: <https://docs.gitlab.com/ce/ssh/README.html> ("Generate an SSH key pair")
- The most important point is that `ssh` is able to find your key pair, i.e., it needs to be located in your HOME folder

If everything works well, you should receive the following friendly welcome message after typing in the command `ssh -T git@github.com:`

Hi berndweiss! You've successfully authenticated, but GitHub does not provide shell access.



# Exercise

- In a previous exercise (see ), you have created your own repository (let's call it `your-new-repo`)
- Now, go to your GitHub account and create a new repository on GitHub
  - Startpage -> tab "Repositories" -> green button "New"
  - Enter a new "Repository name"
  - Make it "Private" (unless you have something important to share)
  - Do **not** check any of the "Initialize this repository with" boxes
  - Hit the green "Create repository" button
  - Choose SSH or HTTPS as protocol ("Quick setup — if you've done this kind of thing before")
  - Look for "...or push an existing repository from the command line"

