Version Control with Git

P. Guttmann

Hochschule Furtwangen

29 Mar 2024

A Sincere Confession

I have been there.

You too? 👀

report.pdf	20.03.2024 01:38	Microsoft Edge PDF Document	0 KB
report_alice.pdf	20.03.2024 01:38	Microsoft Edge PDF Document	0 KB
report_bob.pdf	20.03.2024 01:38	Microsoft Edge PDF Document	0 KB
report_alice_corrected.pdf	20.03.2024 01:38	Microsoft Edge PDF Document	0 KB
report_alice_bob_combined.pdf	20.03.2024 01:38	Microsoft Edge PDF Document	0 KB
report_alice_bob_combined_final.pdf	20.03.2024 01:39	Microsoft Edge PDF Document	0 KB
report_submission.pdf	20.03.2024 01:39	Microsoft Edge PDF Document	0 KB
report_submission_improved.pdf	20.03.2024 01:39	Microsoft Edge PDF Document	0 KB

There is Hope!

- Version Control Systems (VCS)
  - Git
  - Subversion
  - Mercurial
  - Bitkeeper
  - ...

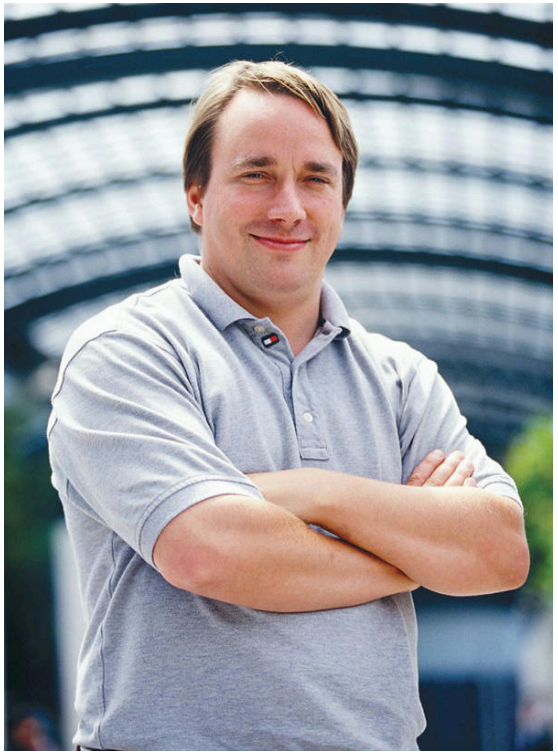
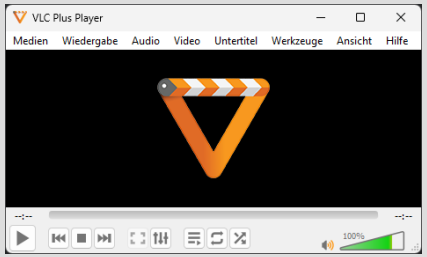


Figure 1: Linus Torvalds[1]

[1] Linuxmag.com, December 2002 https://upload.wikimedia.org/wikipedia/commons/6/69/Linus_Torvalds.jpeg

Online Service	• GitHub • GitLab GitHub	• Youtube • DailyMotion • AtoPlay YouTube
Local Application	• Git git	• VLC Player • Windows Media Player 

Git

- local
- distributed
- command line interface

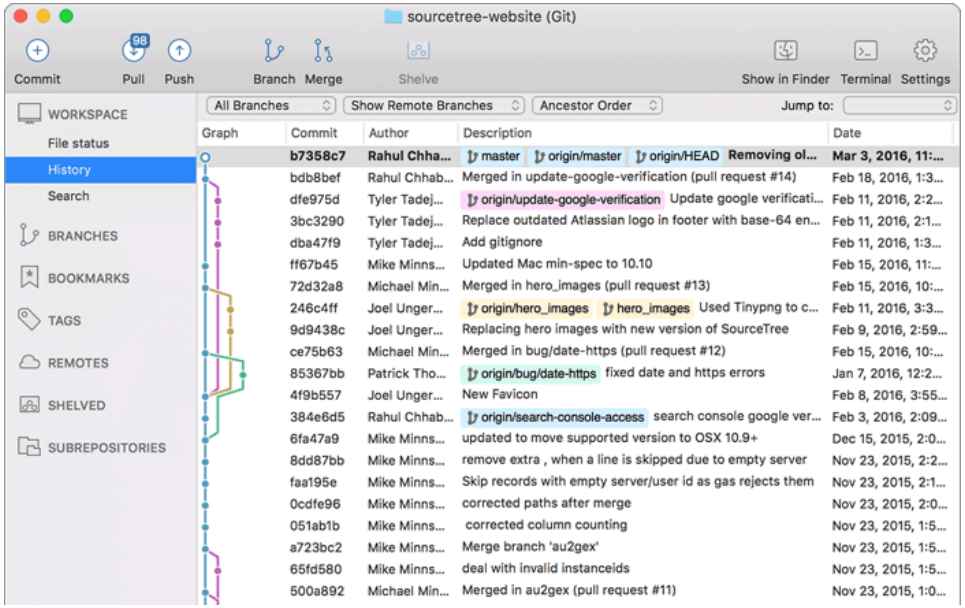
Command Line Interface? Really!?

There are graphical user interfaces.
Choose what you like:

- https://git-scm.com/downloads/guis

Git is also integrated in many IDEs:

- eg. Visual Studio Code[2,3]
  - recommendation: extension: `GitGraph`



[2] https://code.visualstudio.com/

[3] https://code.visualstudio.com/docs/sourcecontrol/intro-to-git

Repository

Repository

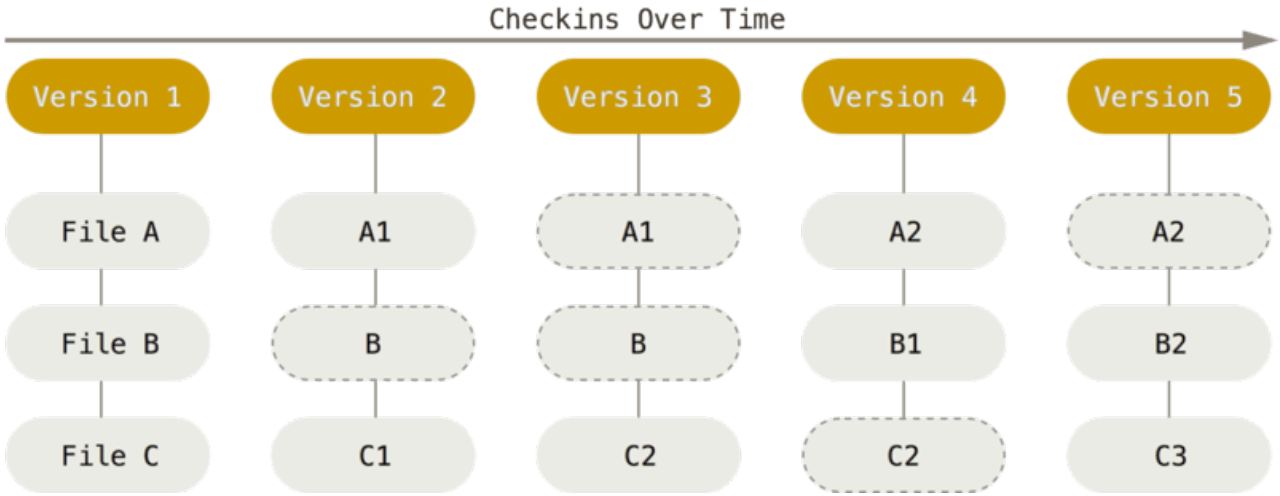
- `.git` directory
  - tracked files
  - all versions of files
  - meta data

```
# initialize a new (empty) repository
git init
```

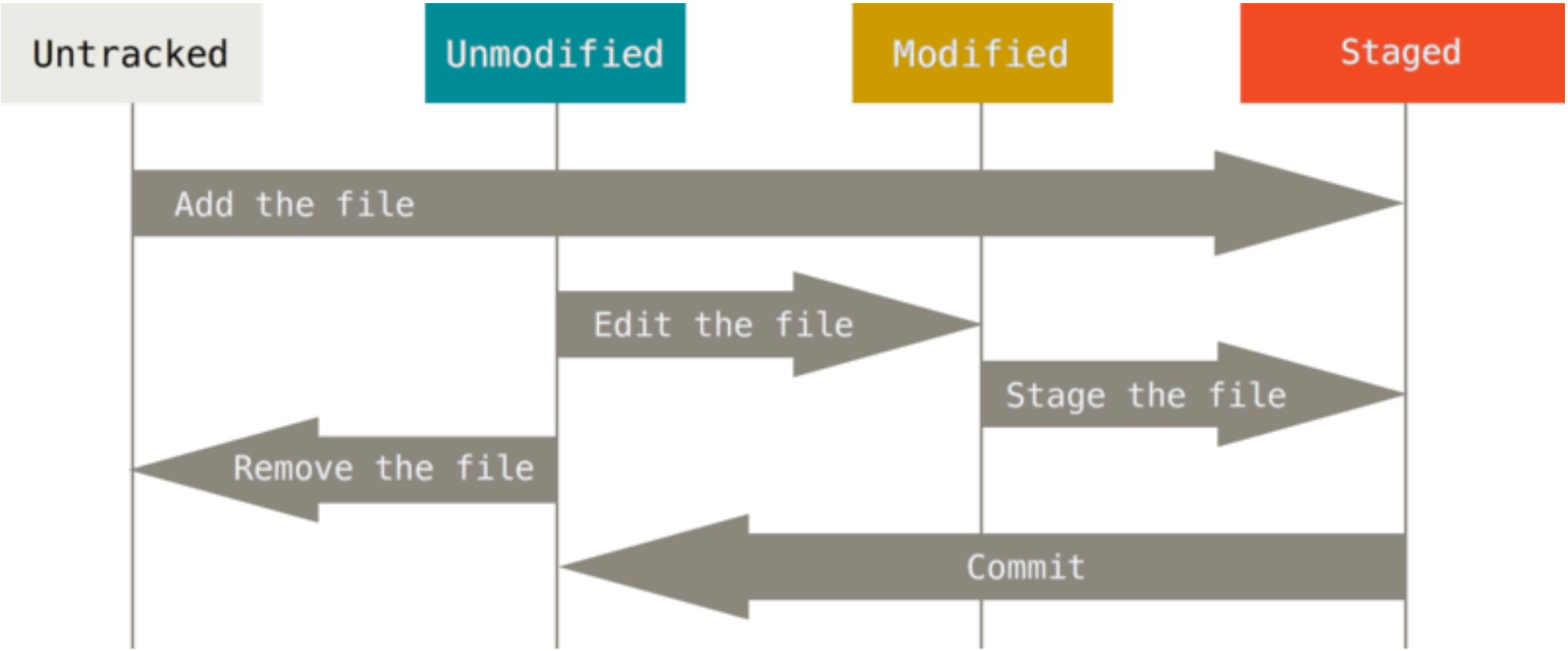
Commits

Commit

- "Version" of files in repo
- Stores `snapshot` of files
- **No** "delta"



Staging and Committing



Staging and Committing


```
git add newFile.txt # adding an untracked file
git add modifiedFile.txt # staging a modified file
git add . # adding / staging all files

git commit -m "My commit message here" # committing
```

Staging and Committing


```
git status # observing the status of the staging area

git diff # differences (modified <-> unmodified)
git diff --staged # differences (staged <-> unmodified)

git log # log of commits
git log --graph # log + "commit graph"
git log --patch # log + "patch" (changes)
```

Live Demo !

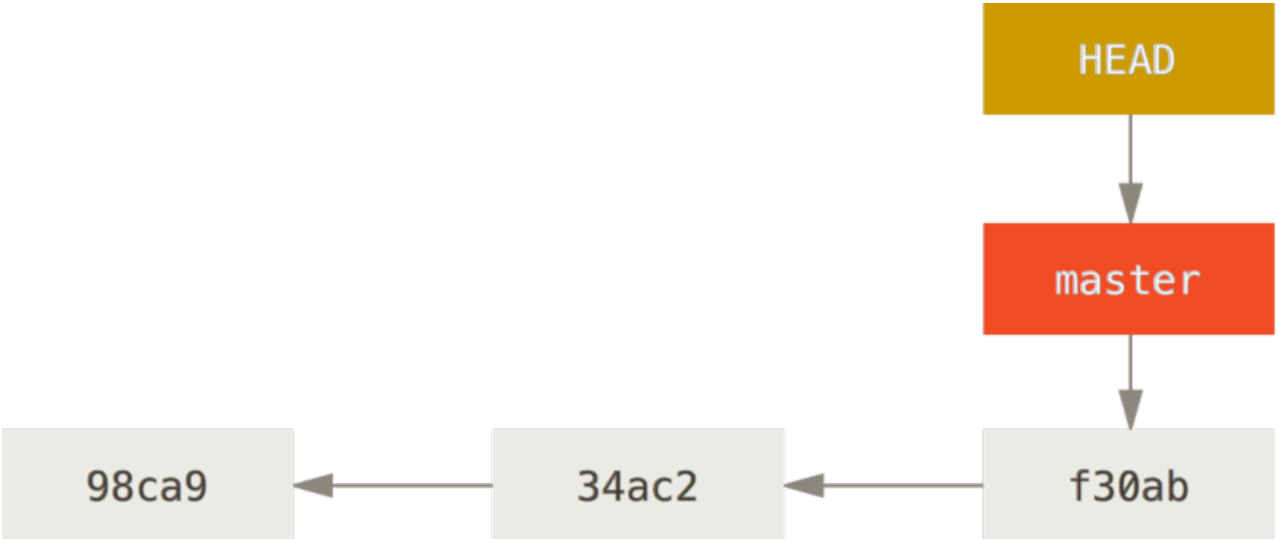
- Poems and Code

Branching

Branch

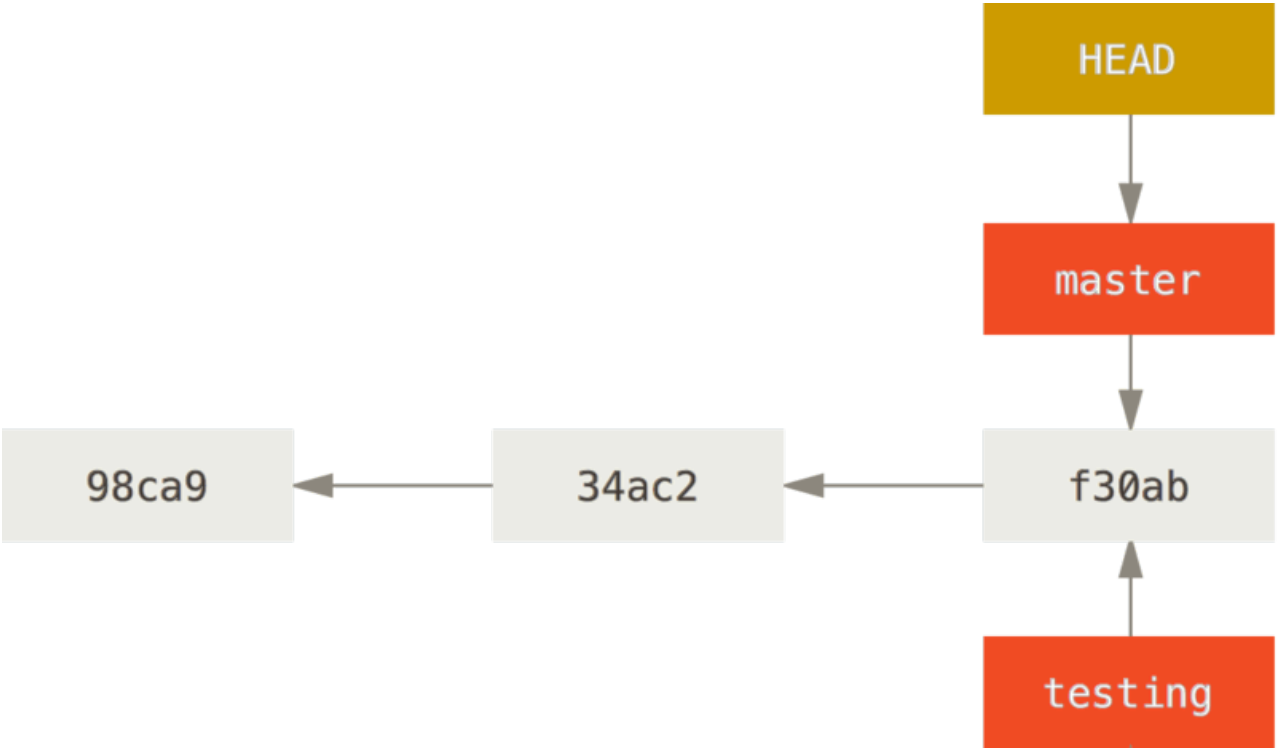
- history of commits
- can be named
- can diverge

```
# view branches
git branch
```



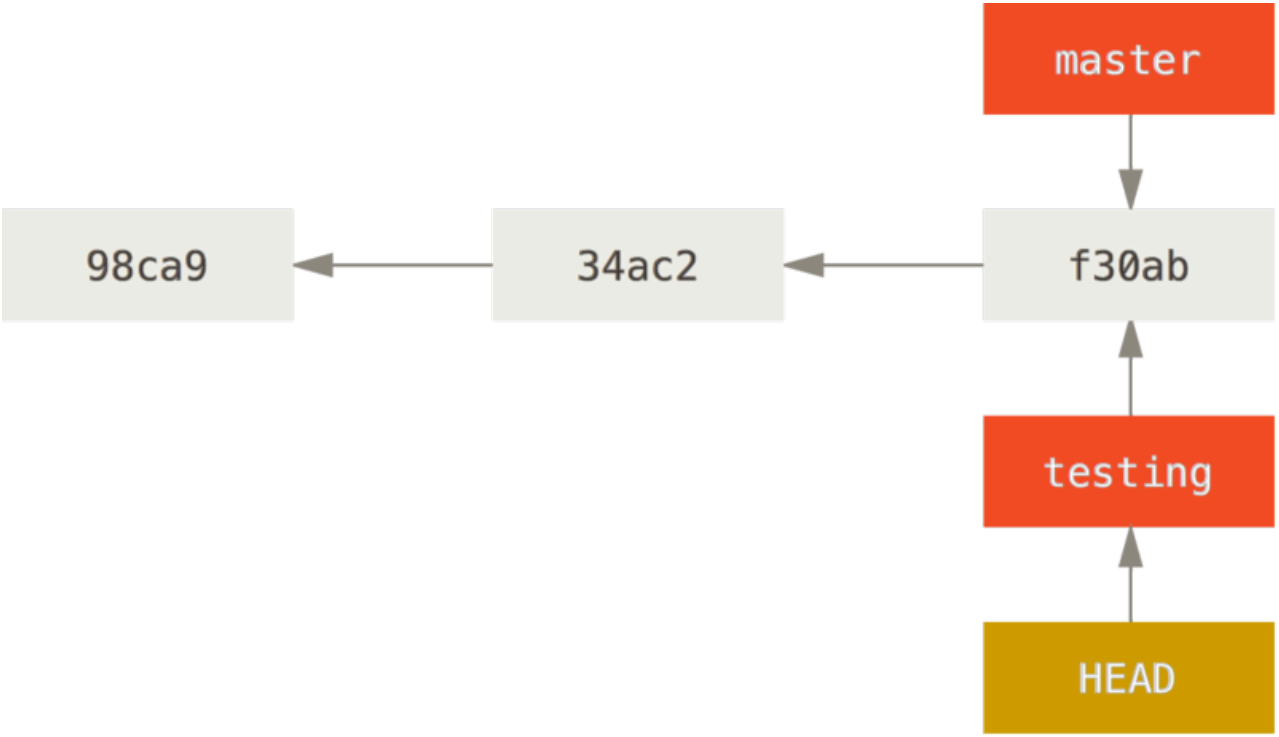
Branching

```
# create branch testing
git branch testing
```



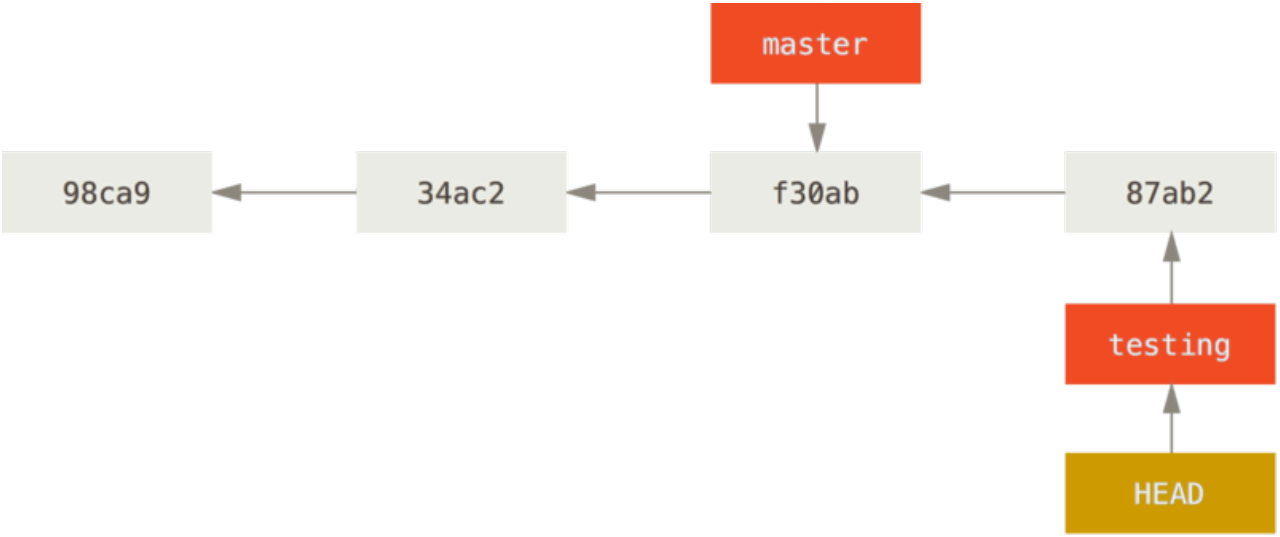
Branching

```
# switch to branch testing
git switch testing
```



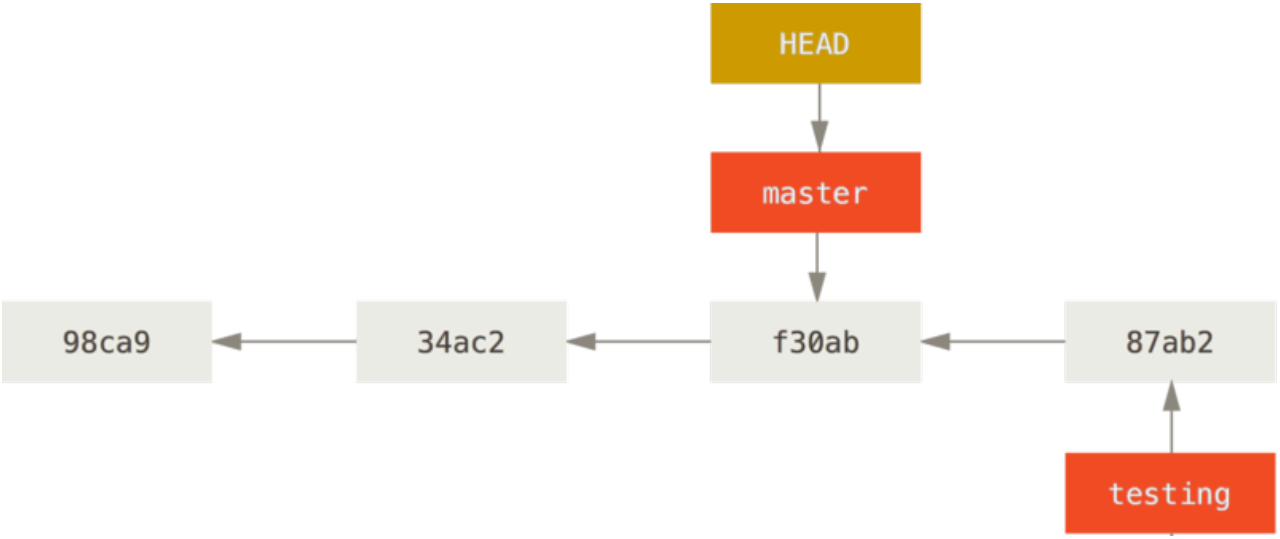
Branching

```
# commit to branch testing
git commit -a -m "My experiments"
```



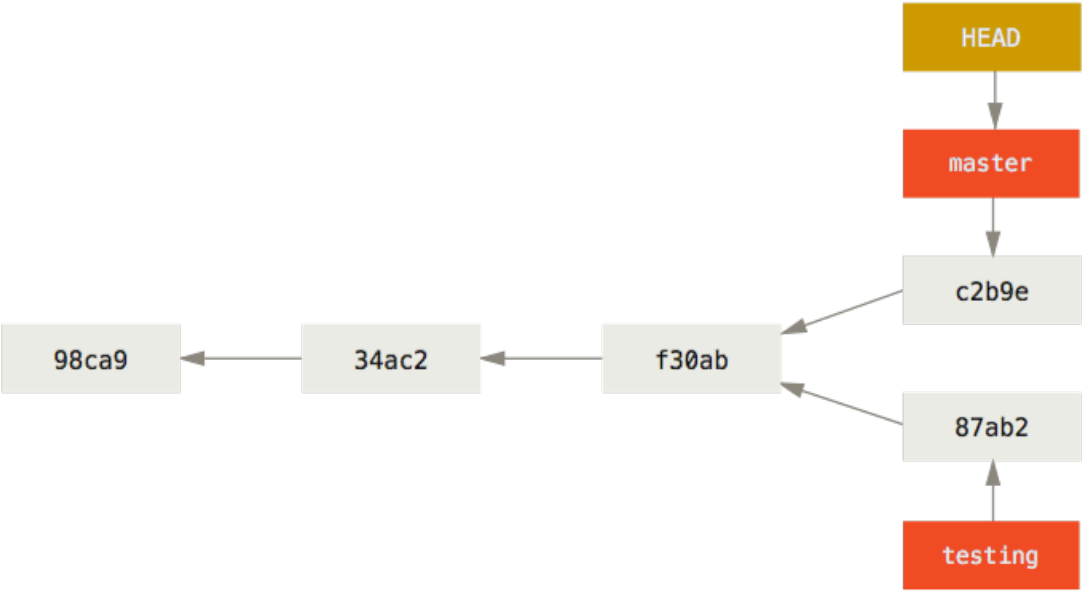
Branching

```
# switch to branch master
git switch master
```



Branching

```
# commit to branch master
git commit -a -m "My other changes"
```



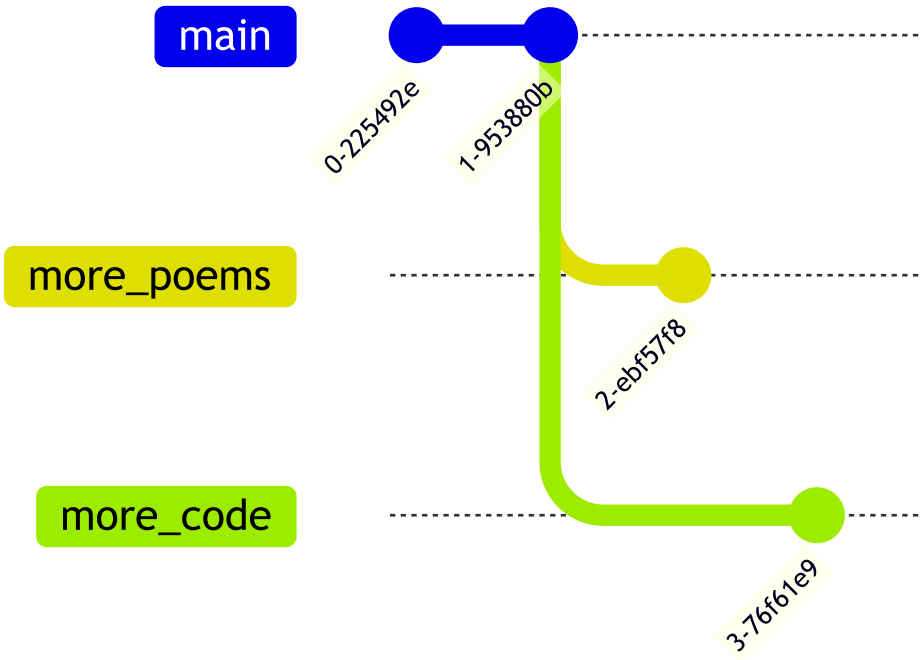
Live Demo !

- More Poems and Code

Merging

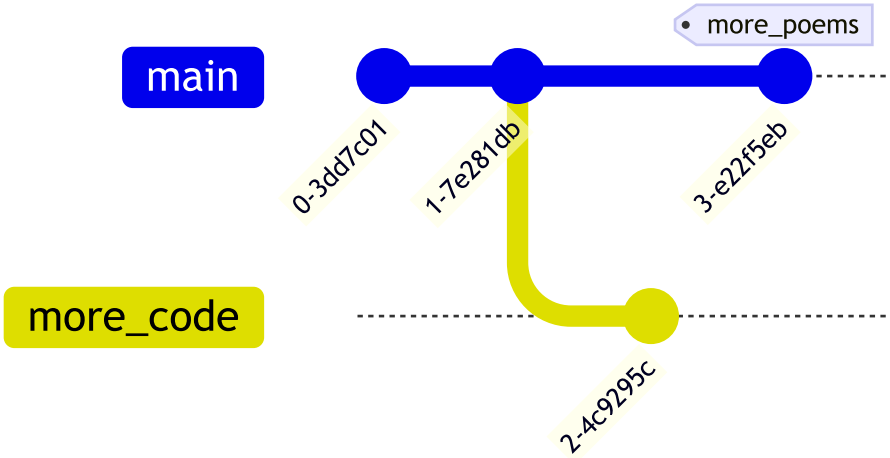
Merge

- combining branches



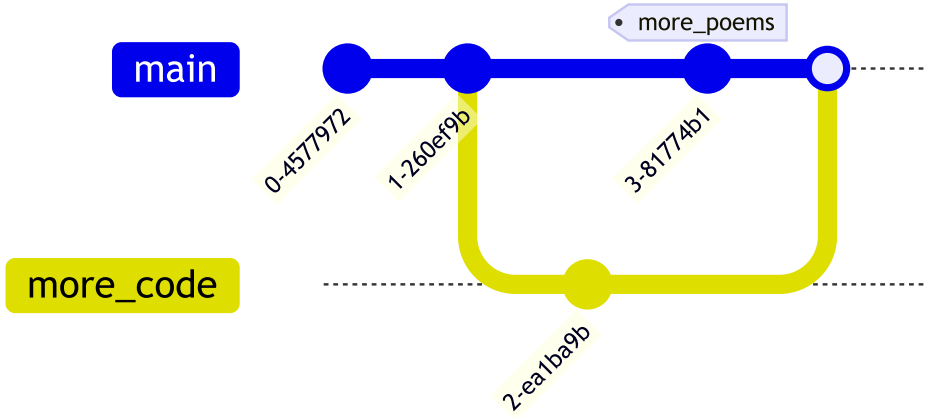
Merging

```
git switch main

# get changes from
'more_peoms' into main
git merge more_poems
```

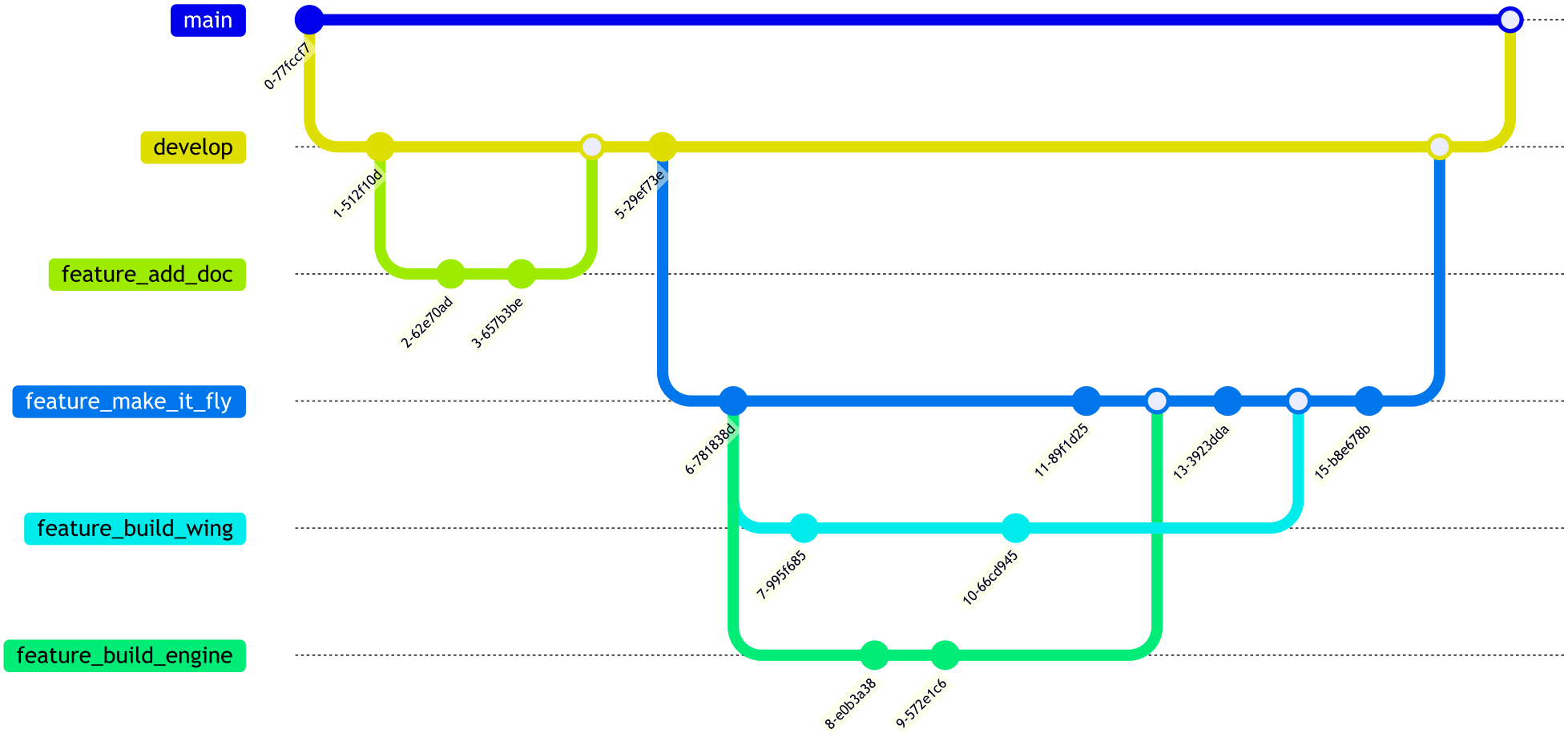


Merging

```
git switch main

# get changes from
'more_code' into main
git merge more_code
```



Gitflow


main
develop
feature_add_doc
feature_make_it_fly
feature_build_wing
feature_build_engine
0-77fccf7
1-512f10d
2-62e70ad
3-657b3be
5-29ef73e
6-781838d
7-995f685
8-e0b3a38
9-572e1c6
10-66cd945
11-89f1d25
13-3923dda
15-b8e678b

Live Demo !

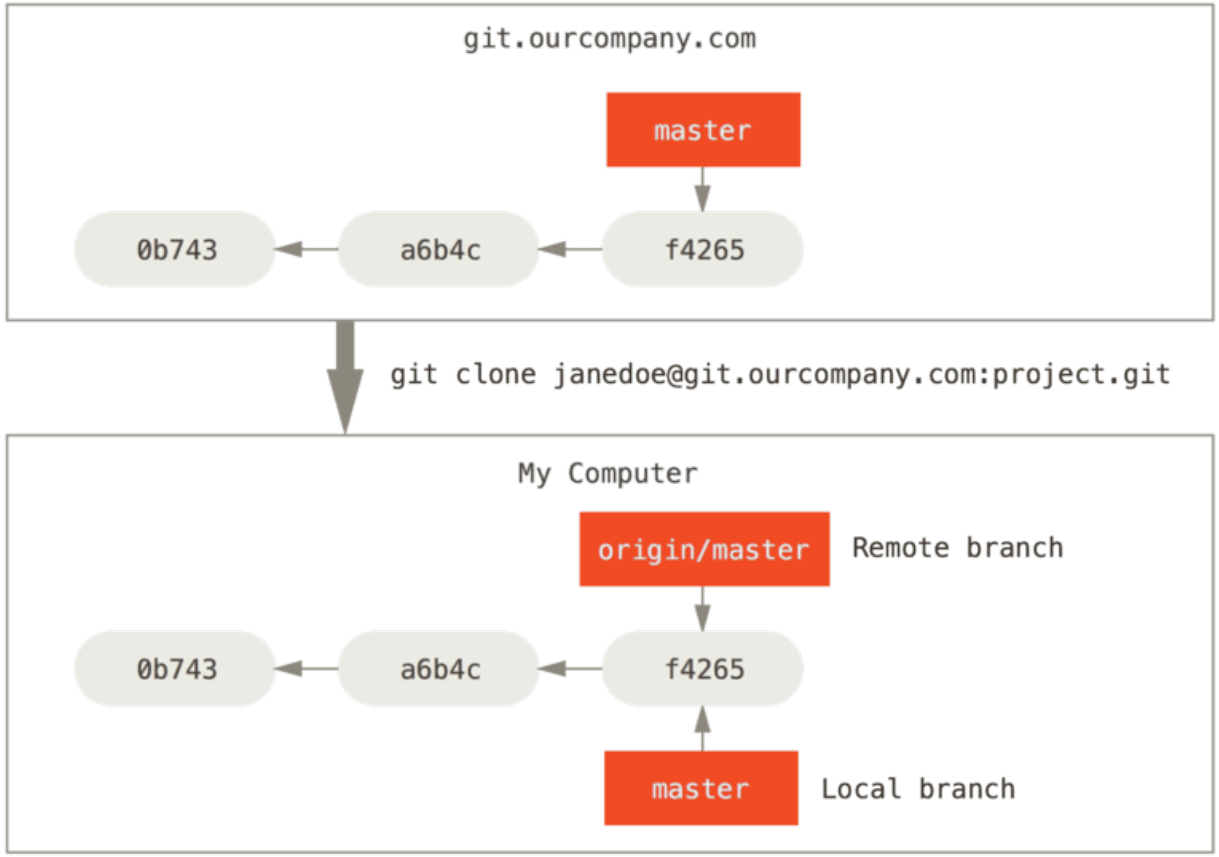
- Merging Poems and Code
- Merge Conflict

Decentralized Multi-Repository Git

Remote

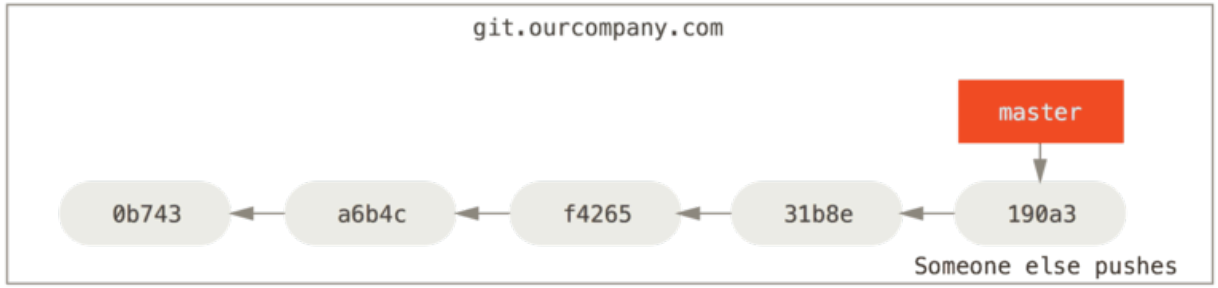
- upstream repository
- external (web or local)

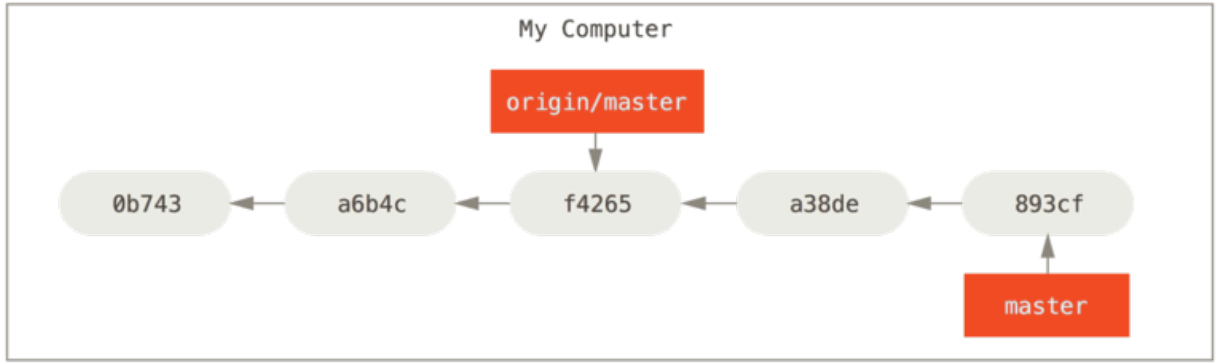
```
# Clone an existing repo
git clone <repository>
```



Decentralized Multi-Repository Git

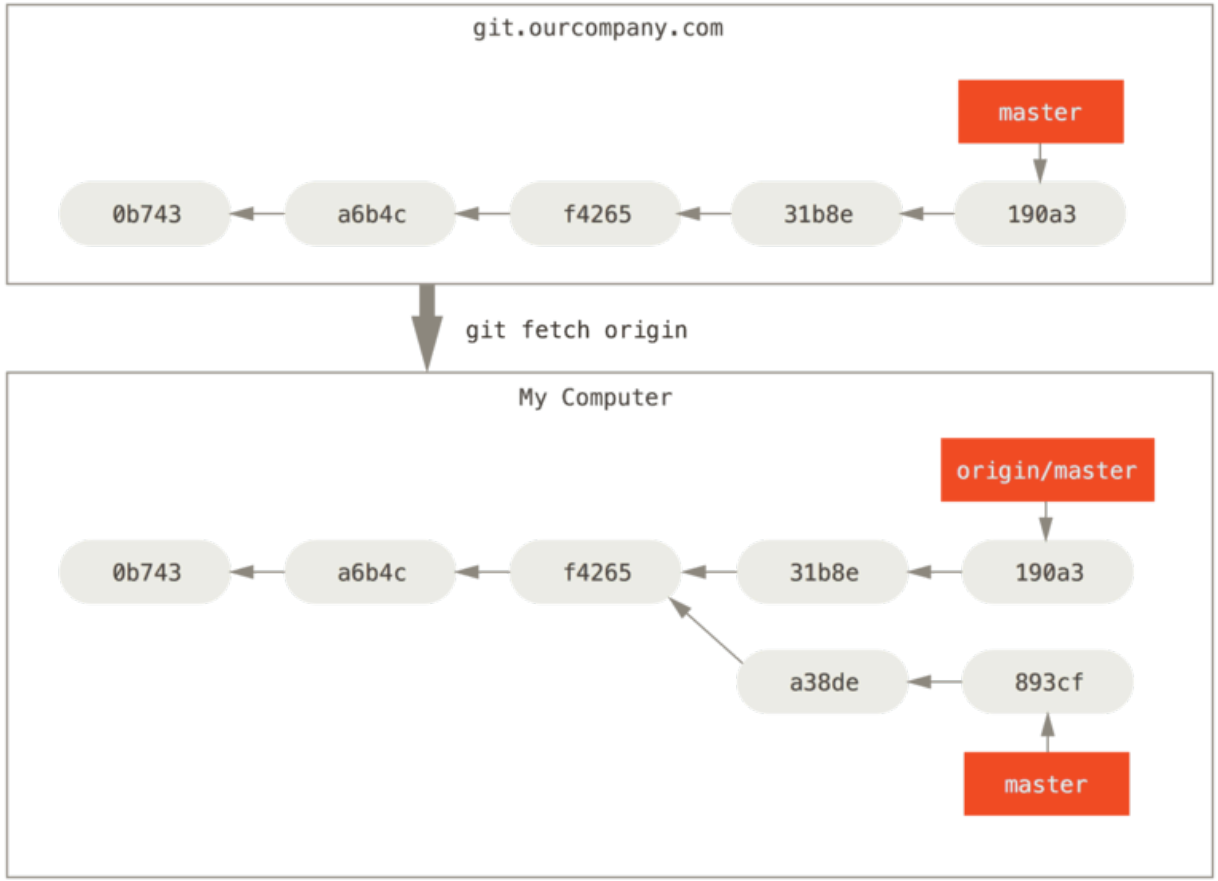
```
# do work
git commit

# colleagues do
work in remote
```





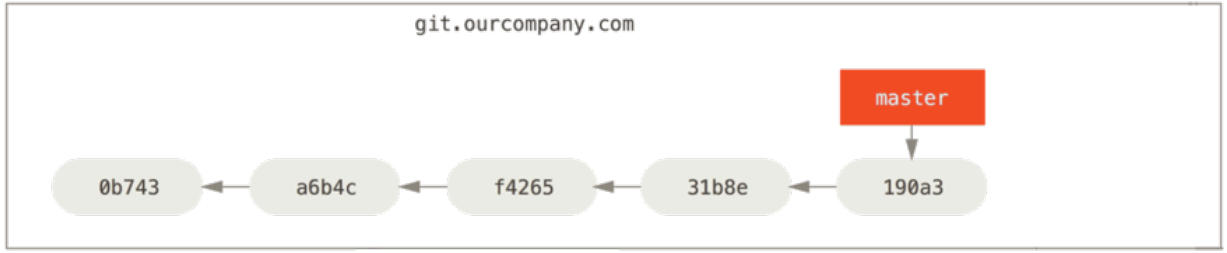
Decentralized Multi-Repository Git

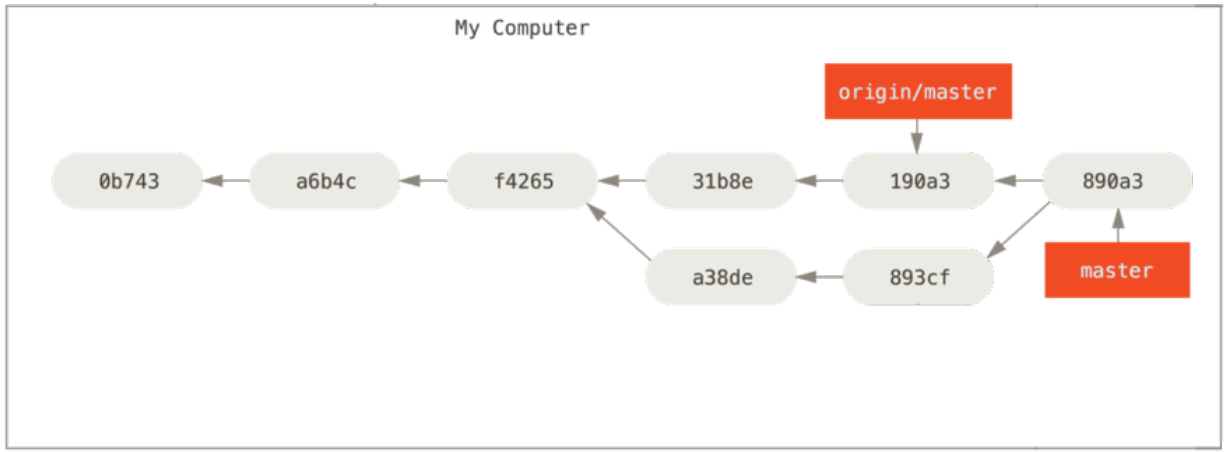

```
# fetch changes
from remote
git fetch origin
```



Decentralized Multi-Repository Git

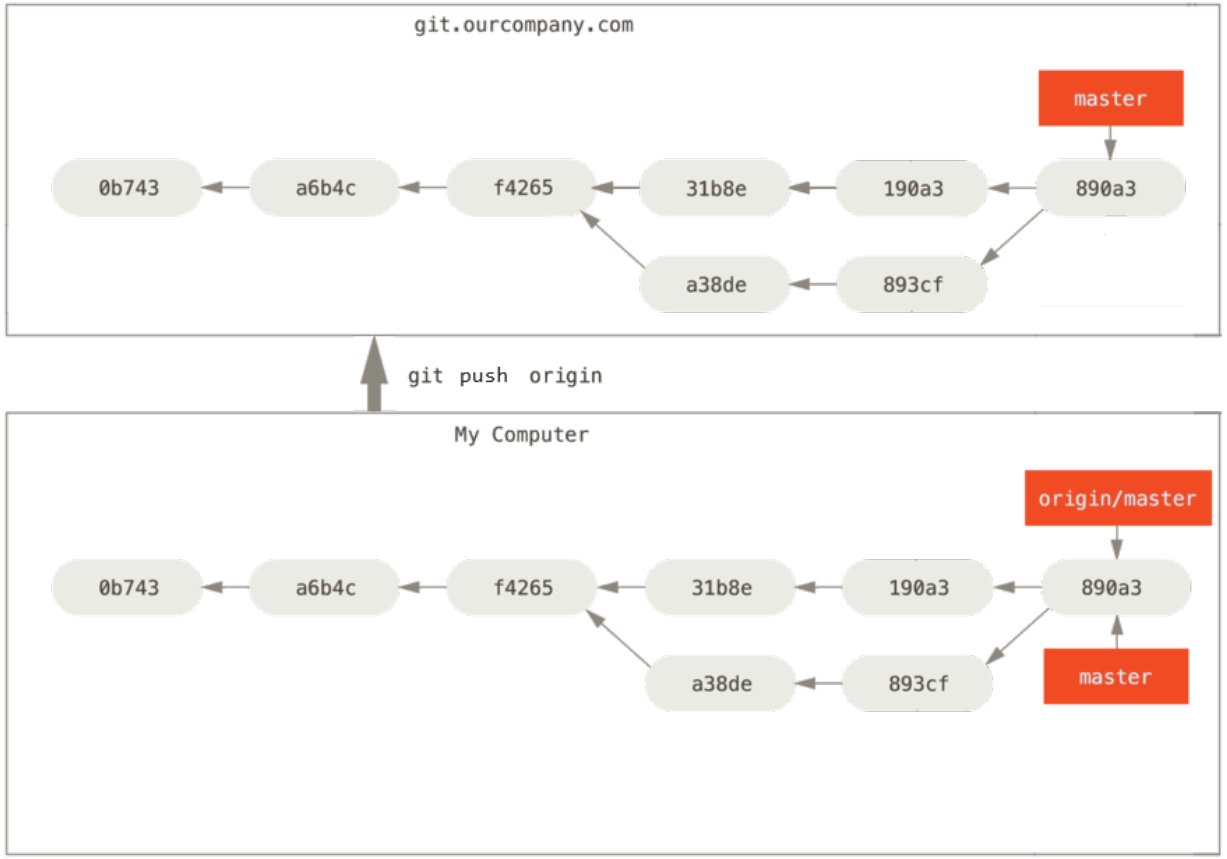
```
# local merge
git merge origin/
master
```





Decentralized Multi-Repository Git

```
# push changes to remote
git push origin
```



Live Demo !

- Repositories for Everyone

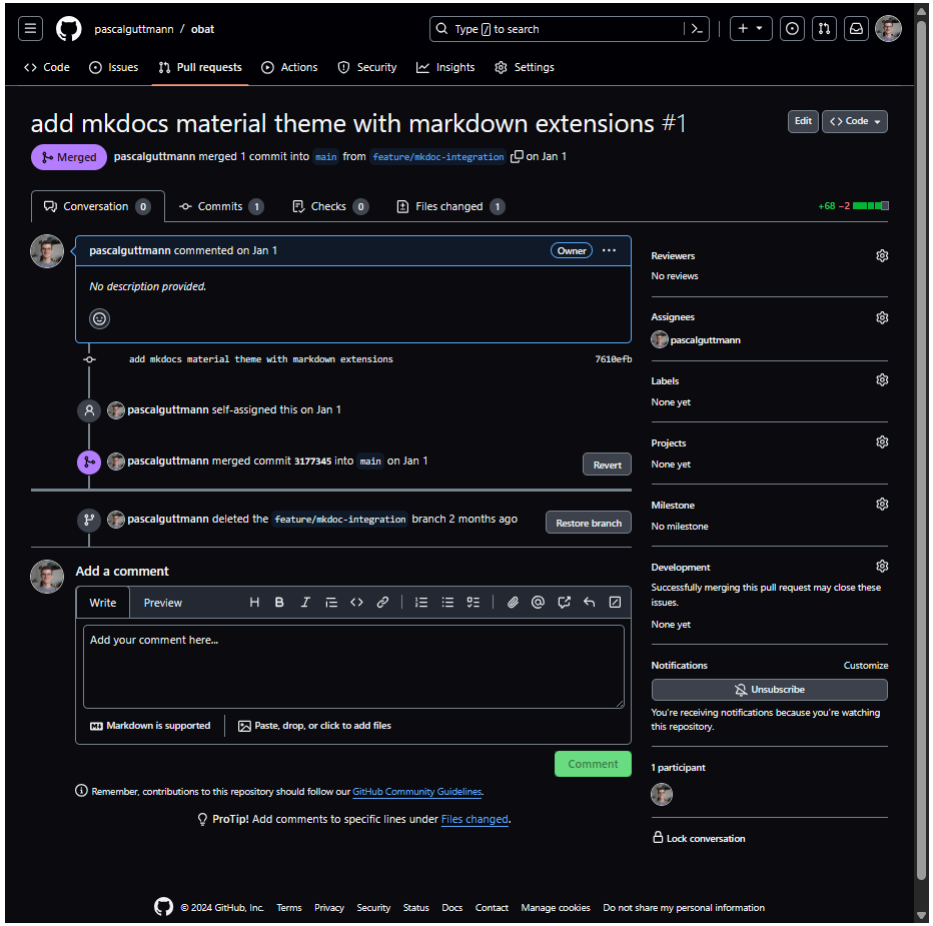
GitHub

- Hosting Git Server Online
- Collaboration Features

Pull Requests

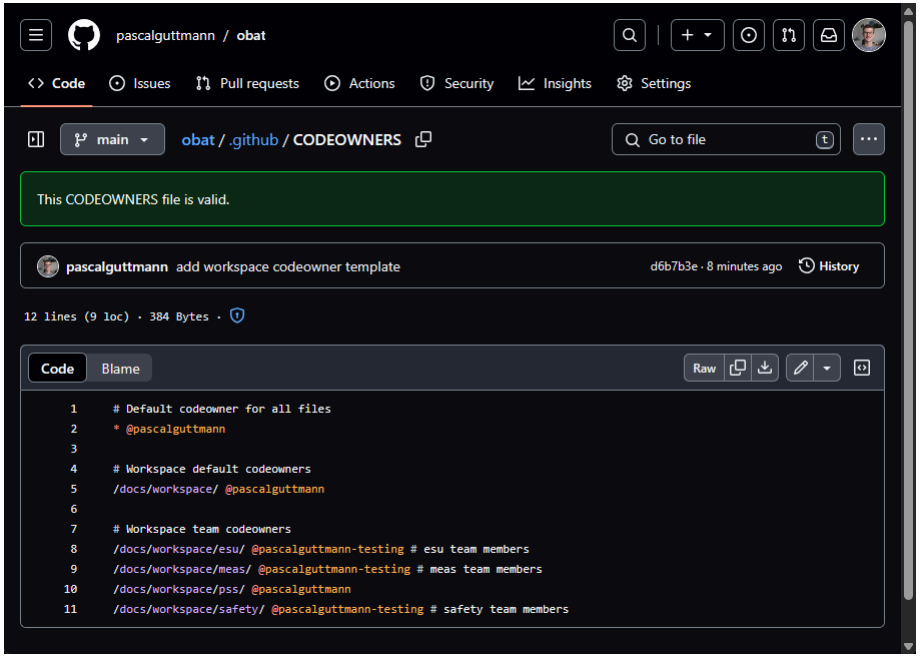
Pull Request

- Request to merge changes
- Review



Code Owners

- define owners of file
- file change: approval of codeowner required



Live Demo !

- GitHub Website
- Creating a Pull Request

Force Pushing

- Rewriting history is kind of lying… Don't do that. At least try to do it locally. (Squashing, etc)
- Force pushing is rewriting history on the remote. This will typically cause a lot of confusion. Don't do that.[4]

[4] Conditions apply: Sensitive data committed, etc.

Data Recovery & Sensitive Information

- When data **is committed**, it is very hard to completely loose it.[5]
- Do not commit sensitive or personal data. If it happens:
  - remove it yourself
  - ask your administrator for help

[5] Advanced rescue strategies might be needed.

Binary Data

- Git itself **can** easily store and merge binary data
  - If not compressible the repository size might increase[6]
  - Merge conflicts **must** be resolved by you! That is very hard for binary data.[7]

[6] Solution: Git Large File System (Git LFS)

[7] External Diff and Mergetools can by utilized if necessary

Installation & Setup

Installing Git

Installation Help:
https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

On Linux (using `apt`):

```
sudo apt install git-all
```

On Windows:
https://git-scm.com/download/win

On Mac OS:
https://git-scm.com/download/mac

Configuration

```
# setup identity
git config --global user.name "Max Mustermann"
git config --global user.email max@domain.de

# avoid hassle exiting vim
git config --global core.editor notepad

# for comfort
git config --global push.default current
git config --global push.autoSetupRemote true
git config --global push.followtags true
```

Configuration

```
# use ssh for GitHub (avoid passwords -> ssh keypair)
git config --global url.ssh://git@github.com/.insteadof
https://github.com/


# alias `git lg` to print a fancy log
git config --global alias.lg "log --graph --abbrev-commit
--decorate --format=format:'%C(bold blue)%h%C(reset) -
%C(bold cyan)%ah%C(reset) %C(bold green)(%ar)%C(reset)
%C(white)%s%C(reset) %C(dim white)- %an%C(reset)%C(bold
yellow)%d%C(reset)' --all"
```

Setting up GitHub

1. Sign up at: (Select free plan)

https://github.com/

2. Sign into your newly created account:

https://github.com/login

3. Generate ssh key:

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Setting up GitHub

4. Set up ssh keys:

https://docs.github.com/de/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

5. Test the authentication:

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/testing-your-ssh-connection

Generate SSH key

Start `Git Bash` Run command:

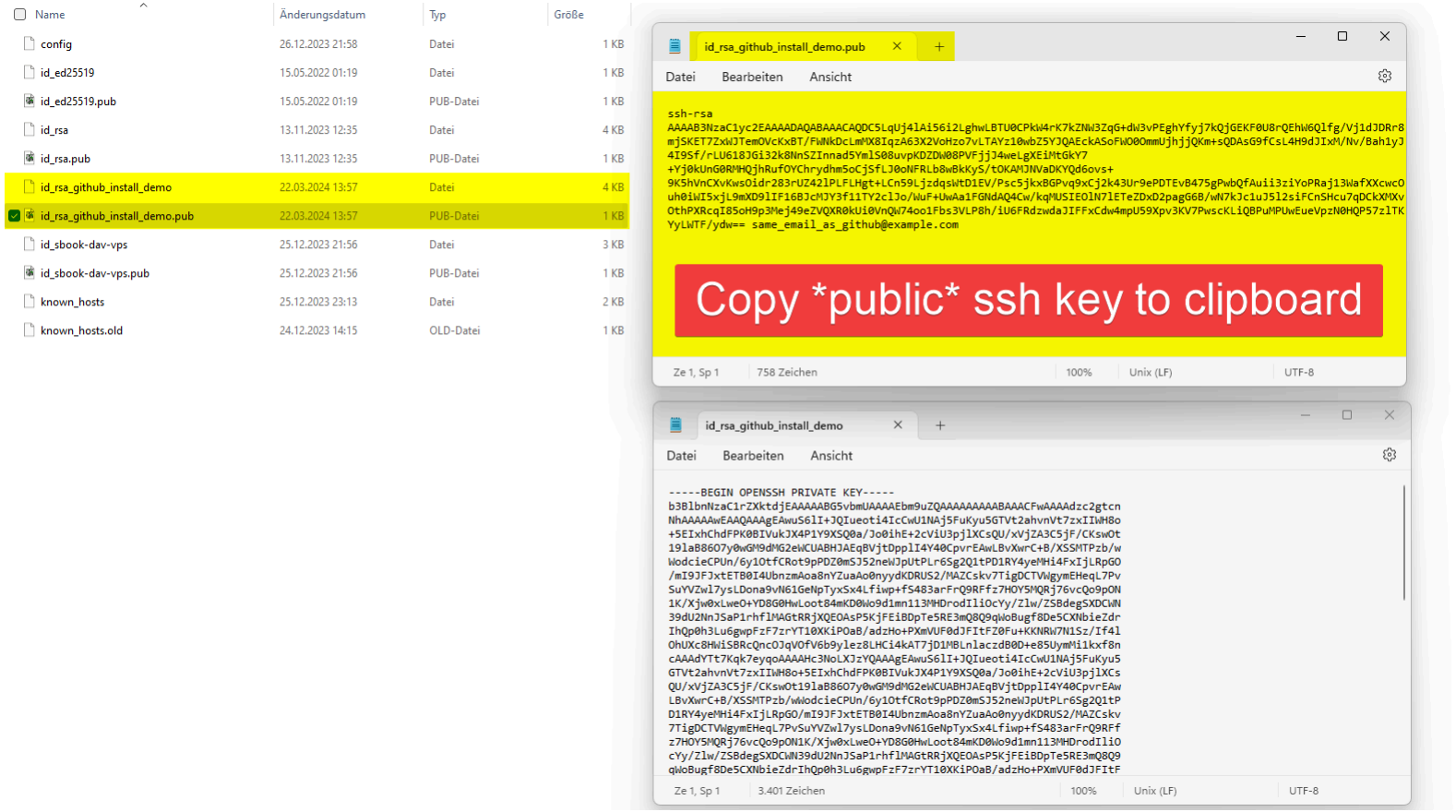
```
ssh-keygen -t rsa -b 4096 -C
"same_email_as_github@example.com"
```

If prompted for file path: Specify or use default with enter. If prompted passphrase: Hit enter. (Confirm: enter again).

Setting up GitHub


```
Pascal_Guttmann@DESKTOP-022I5MO MINGW64 ~/Documents
$ ssh-keygen -t rsa -b 4096 -C "same_email_as_github@example.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Pascal_Guttmann/.ssh/id_rsa): /c/Users/Pascal_Guttmann/.ssh/id_rsa_github_install_demo
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Pascal_Guttmann/.ssh/id_rsa_github_install_demo
Your public key has been saved in /c/Users/Pascal_Guttmann/.ssh/id_rsa_github_install_demo.pub
The key fingerprint is:
SHA256:4bz1PZIHFDiBxQJKJbWa95bLYbXKfQLMBhJ94i+a55o same_email_as_github@example.com
The key's randomart image is:
+---[RSA 4096]----+
|    o++. +oo.    |
|   ..oo.+ +  .   |
|    .o.o.. ..    |
|    .ooo . .     |
|    o..=S o .    |
|     ...*= o +   |
|     o oB.. + +  |
|    o..= =. .o . |
|    E+. = .o     |
+----[SHA256]-----+
```

Setting up GitHub

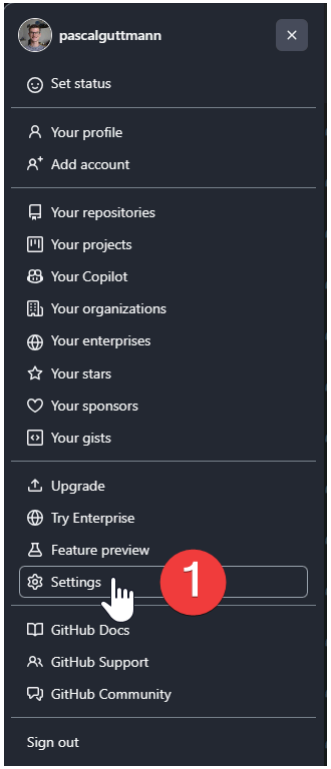


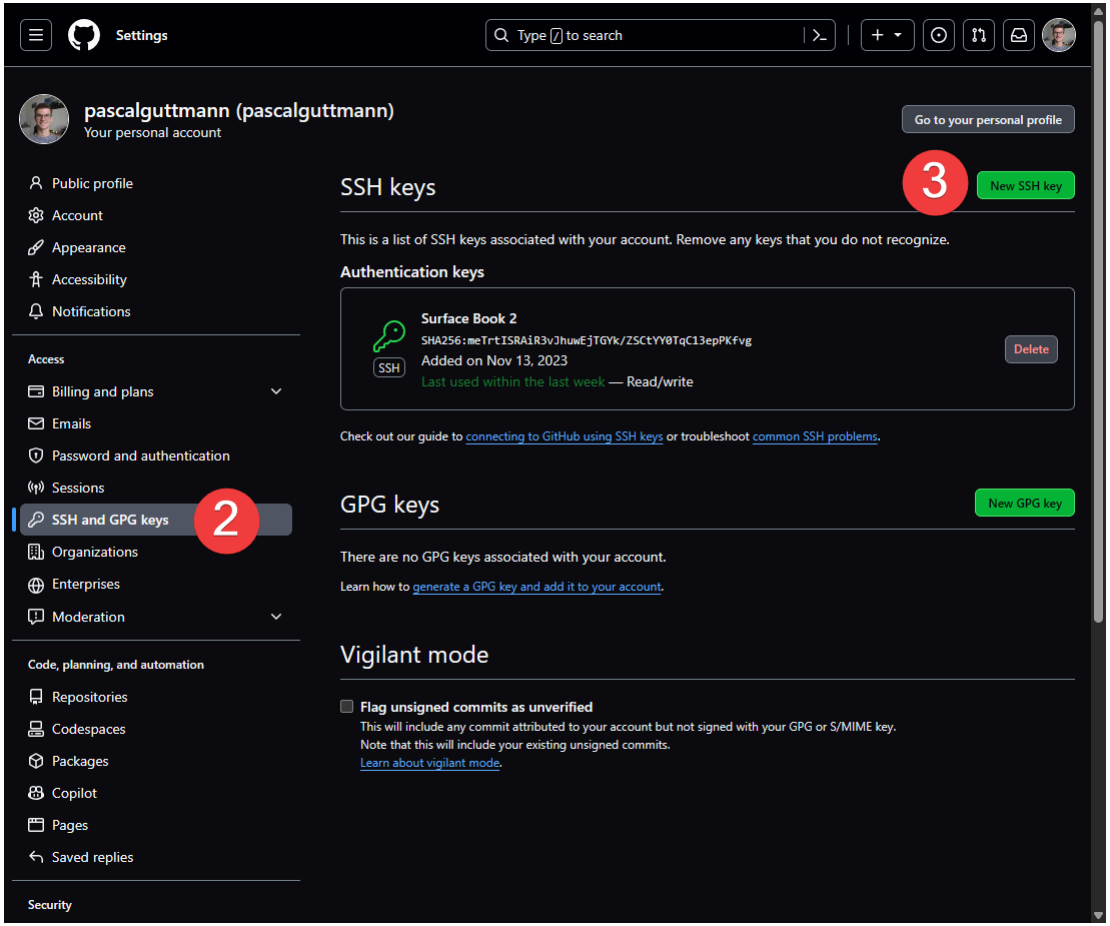
Setting up GitHub

Set up SSH key in GitHub

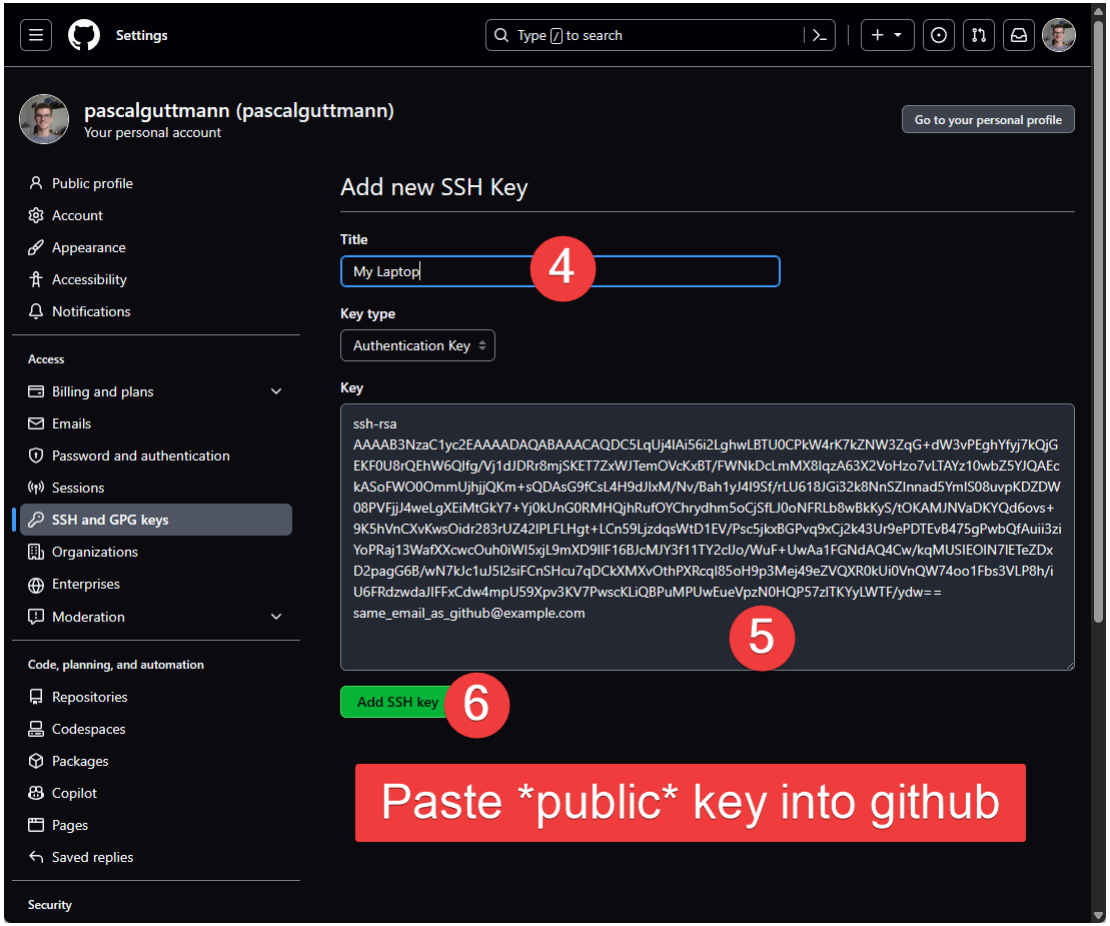
1. Settings
2. SSH and GPG Keys
3. New SSH key
4. Give name to key
5. Paste public SSH key
6. Confirm

Setting up GitHub





Setting up GitHub



Test the authentication

```
ssh -T git@github.com
```

If “authenticity cannot be established, are you sure you want to continue?” Type `yes` and hit enter.

Getting Help

Getting Help

1. https://git-scm.com/book/en/v2
2. `git <command> --help # by default opens online help`
3. https://docs.github.com/en/get-started/start-your-journey
4. `git help # show commandline help`
5. https://training.github.com/downloads/github-git-cheat-sheet.pdf