Juniorprogrammierer.de

Java 1.5: Github

2024/25 – Sascha Stojanovic

Agenda

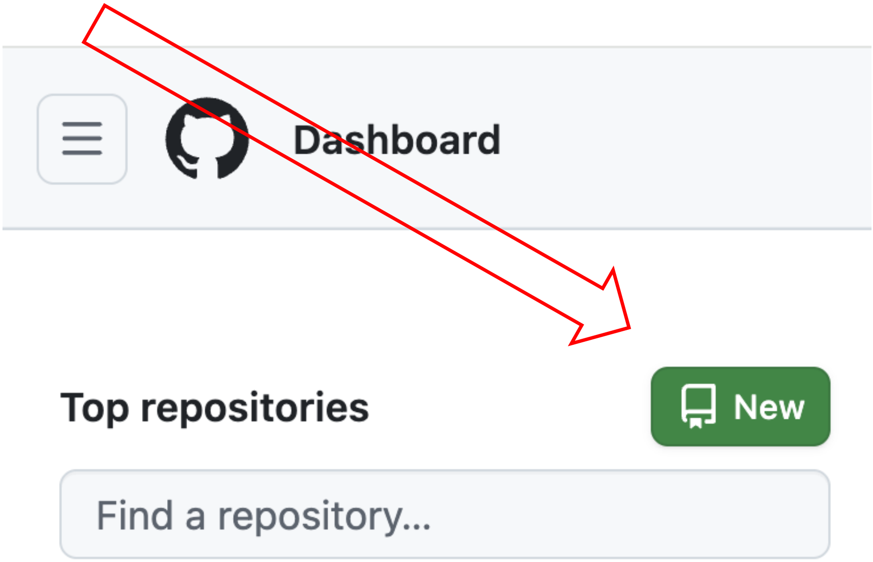
- What problems does github.com solve?
- Getting started with github I to V
- Important buzzwords I to IV
- Lets do some exercises!

What problems does github.com solve

- Versioning
  - it documents the history of your coding
  - You can jump back to any coding version you like
- Distribution
  - You can simply git clone a repo to get the code of someone else
  - By providing you git repo link you can forward your project
- Standardization
  - Widely used in communities and companies
- Team work
  - A common platform for you and your team to code on the same project

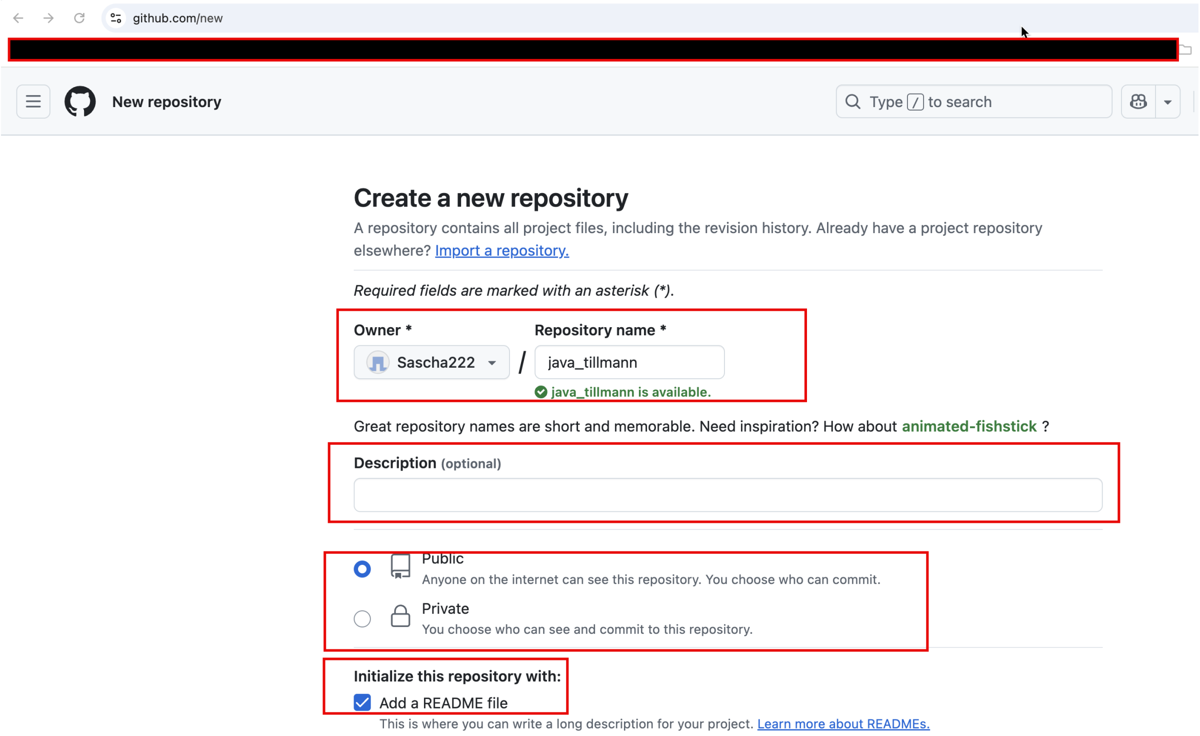
Getting started with github I

- Sign up for github.com
- Create a repository



Getting started with github II

- Create a repository II



github.com/new

[redacted]

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * Sascha222 / **Repository name *** java_tillmann

✓ java_tillmann is available.

Great repository names are short and memorable. Need inspiration? How about animated-fishstick ?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

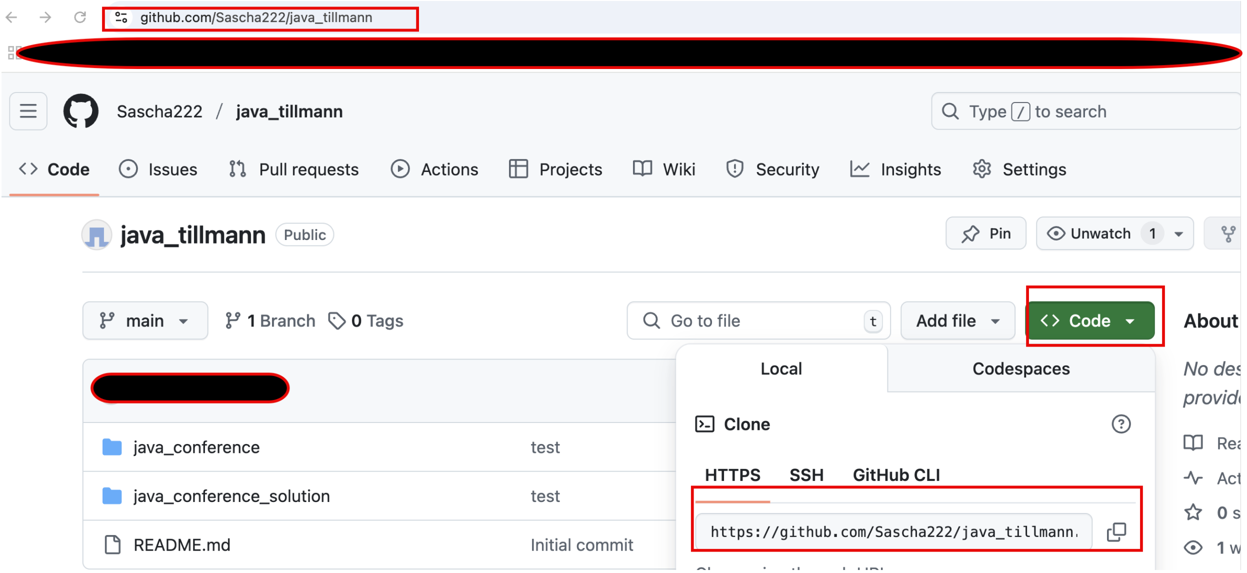
○ Private
You choose who can see and commit to this repository.

Initialize this repository with:

☑ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Getting started with github III

- Open “cmd” and navigate via “cd” to the directory with your java project
- Get the link to your repository



github.com/Sascha222/java_tillmann

[redacted]

Sascha222 / java_tillmann

java_tillmann Public

[redacted]

java_conference	test
java_conference_solution	test
README.md	Initial commit

Clone

HTTPS SSH GitHub CLI

https://github.com/Sascha222/java_tillmann.

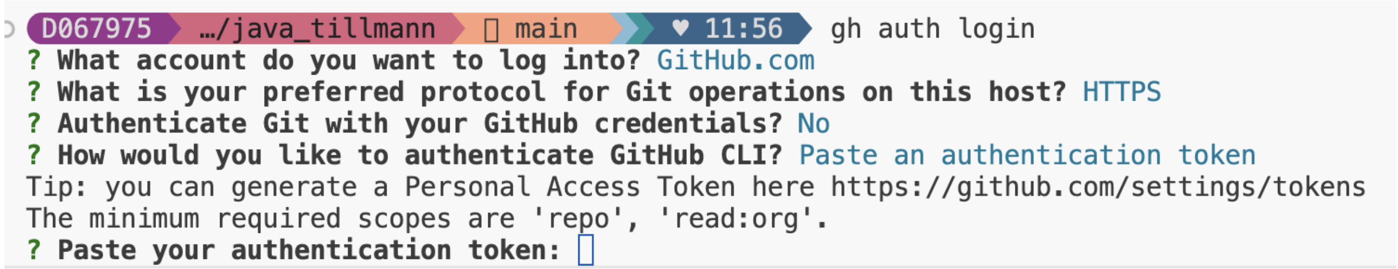
Getting started with github IV

- In your “cmd” (or terminal) enter git clone “your-link”
- A new folder appears with your repo name
- Copy and paste all of you java files into this new folder
- Open the folder with vscode

- How do we get now all of our files into our repository online?

Getting started with github V

- Create online a accesstoken https://github.com/settings/tokens
- Next enter in your console “gh auth login”
- Follow the instructions:



```
 D067975  …/java_tillmann  ▯ main  ♥ 11:56  gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTPS
? Authenticate Git with your GitHub credentials? No
? How would you like to authenticate GitHub CLI? Paste an authentication token
Tip: you can generate a Personal Access Token here https://github.com/settings/tokens
The minimum required scopes are 'repo', 'read:org'.
? Paste your authentication token:
```

- Now enter in the terminal “git status”, then “git add .” , then git commit –m “My first commit”, and last “git push”
- You local vscode is now connected to github.com ☺

Important buzzwords I

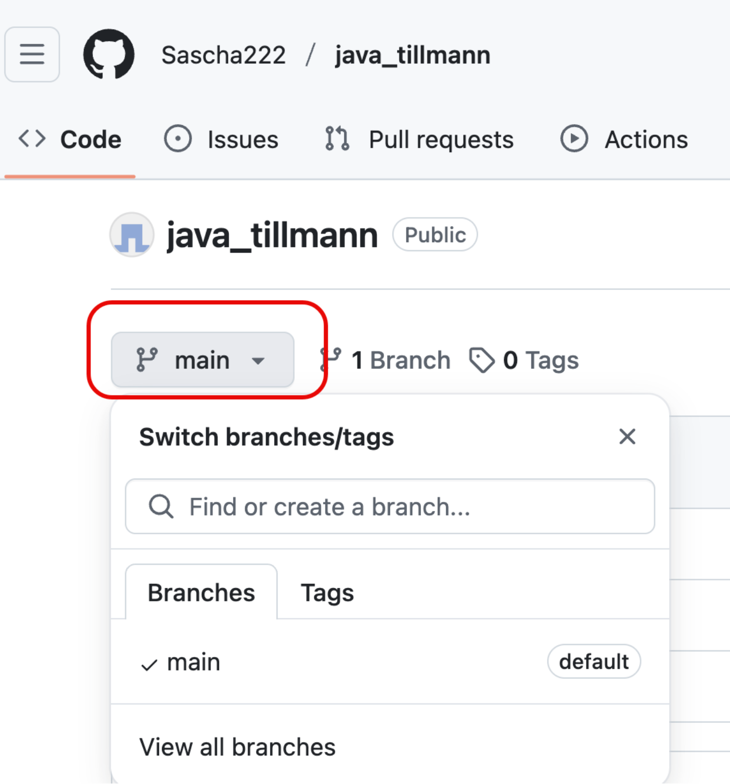
Repository:

- This is your whole project located on github.com
- Your vscode is pushing new code to your repository online or pulling the latest status
- Not only code also other files can be pushed to your repository online
- Try it out:
  - Create a word document called ”test”
  - Enter it in your local repository folder
  - Push your change to your repository
  - Then delete it again

Important buzzwords II

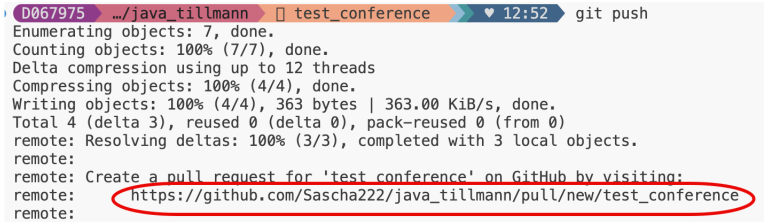
Branch:

- In a team project you are usually never directly push changes into the main-branch => this is the common source of truth
- You create a branch, push it, request review and merge the branches
- Lets do it:
  - Git pull the latest status
  - Create a branch => “git branch branchname”
  - Go to this branch => “git checkout branchname”
  - Now push it to your repo => git add ., git commit –m “name”, git push
- Hold on what is this link?!

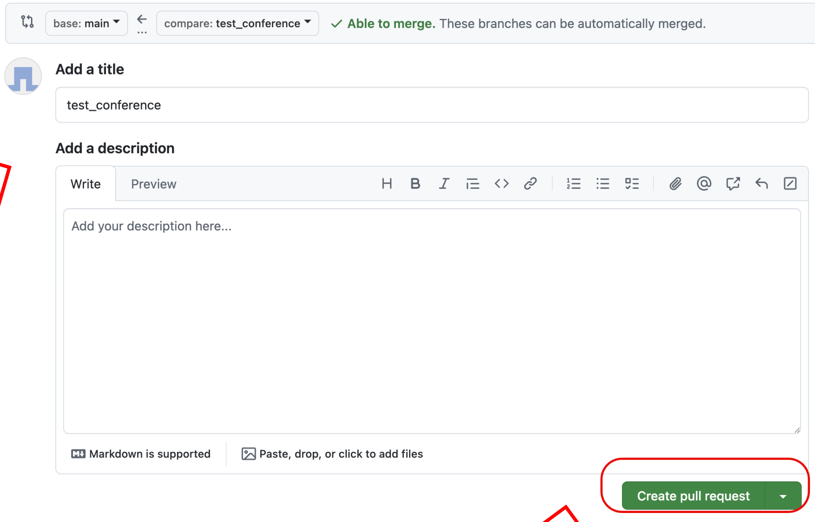


Important buzzwords III

Pullrequest:



```
D067975  …/java_tillmann   test_conference   ♥ 12:52  git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 363 bytes | 363.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote:
remote: Create a pull request for 'test_conference' on GitHub by visiting:
remote:      https://github.com/Sascha222/java_tillmann/pull/new/test_conference
remote:
To https://github.com/Sascha222/java_tillmann.git
 * [new branch]      test_conference
branch 'test_conference' set up to t
```



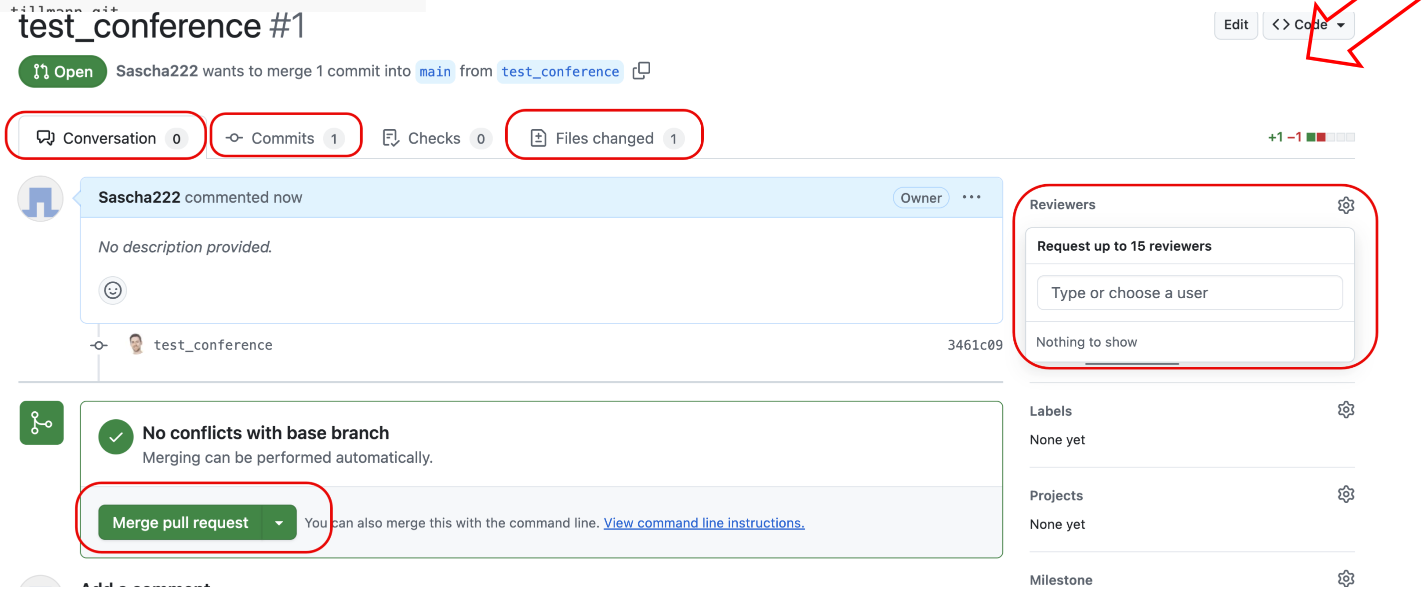
base: main ▾ ← compare: test_conference ▾ ✓ Able to merge. These branches can be automatically merged.

Add a title

test_conference

Add a description

Add your description here...



test_conference #1

Open **Sascha222** wants to merge 1 commit into `main` from `test_conference`

Sascha222 commented now — Owner

No description provided.

test_conference 3461c09

Reviewers

Request up to 15 reviewers

Type or choose a user

Nothing to show

No conflicts with base branch

Merging can be performed automatically.

You can also merge this with the command line. View command line instructions.

Labels

None yet

Projects

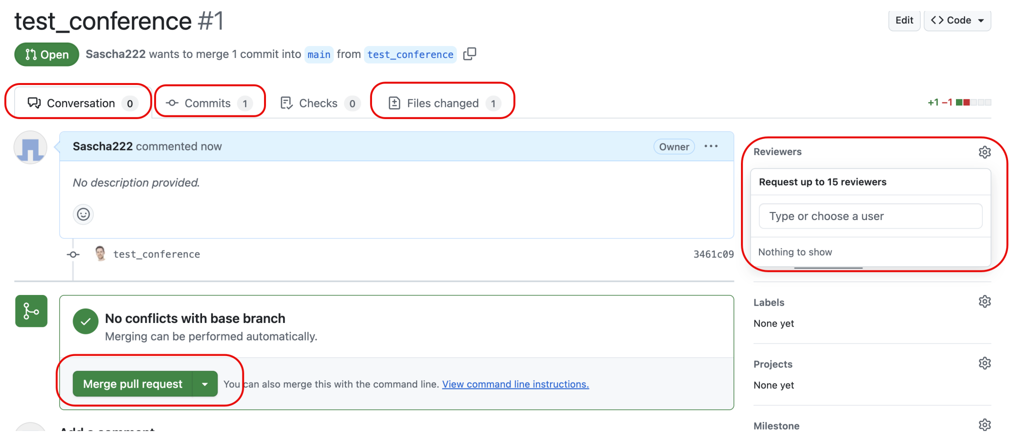
None yet

Milestone

Important buzzwords IV

Pullrequest:

- The first step is to assign a reviewer to your PR
- The reviewer is informed via mail and either approves or request changes with comments
- In case of changes you need to resolve them and request another review
- Once aligned you can merge your branch with the main-branch



test_conference #1

Open Sascha222 wants to merge 1 commit into `main` from `test_conference`

Sascha222 commented now

No description provided.

`test_conference` `3461c09`

No conflicts with base branch

Merging can be performed automatically.

You can also merge this with the command line. View command line instructions.

Lets do some exercises!

- Scenario I: You work solo (for our lesson 90% of the time)
  - Pull the latest status of your main-branch
  - Stay in your main-branch
  - Do any change
  - Push it
- Scenario II: You work in a team and add coding (99% of the time in your daily work)
  - Pull the latest status of your main-branch
  - Create your branch
  - Do any change
  - Push it
  - Create Pullrequest
  - Assign reviewer
  - Do addings (if required) and merge branches
- Scenario III: You are the reviewer (also plenty of times)
  - Someone request a change for main-branch in your repo (you are informed via mail)
  - Leave a comment
  - Comment is solved (you are informed via mail)
  - Approve changes
  - I merge