

# EINFÜHRUNG IN DIE PROGRAMMIERUNG

**VERSION CONTROL WITH GIT (FIRST STEPS)**

**DHBW MANNHEIM**

**WIRTSCHAFTSINFORMATIK (DATA SCIENCE)**

Markus Menth

Martin Gropp

# GIT: VERSION CONTROL

## What is git good for?

- Backup working versions of your code
- Keep a history of your source code
- Collaborate with other programmers

## Couldn't [Dropbox/OneDrive/Google Drive/iCloud] do the same?

- When you're working alone, in many cases: yes.
- But:
  - git helps you to archive a history of your code in consistent, meaningful steps
  - git is a very powerful tool for collaboration in complex projects (originally developed by Linus Torvalds for the Linux kernel)

# TYPICAL GIT USAGE

- create a repository, e.g. on GitLab or GitHub
- ...or fork somebody else's repository
- **clone** the repository on you local computer
- **add** files and **commit** changes
- **push** changes to the remote repository


and:




- **pull** other people's changes from the remote repository


# FORKS


A **fork** is a copy of a repository. It is sometimes used to continue the development of a project separately, sometimes to work on new contributions that can then be merged back into the original repository.


# GITHUB: FORK





[Pulls](#) [Issues](#) [Marketplace](#) [Explore](#)   



 **mgropp / snake** Public

 Unwatch 1


 Star 0


 Fork 0


[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) 



 master 


Go to file


Add file 

Code 

About 

 **mgropp** initial 


on Oct 24  2

 .gitignore

initial

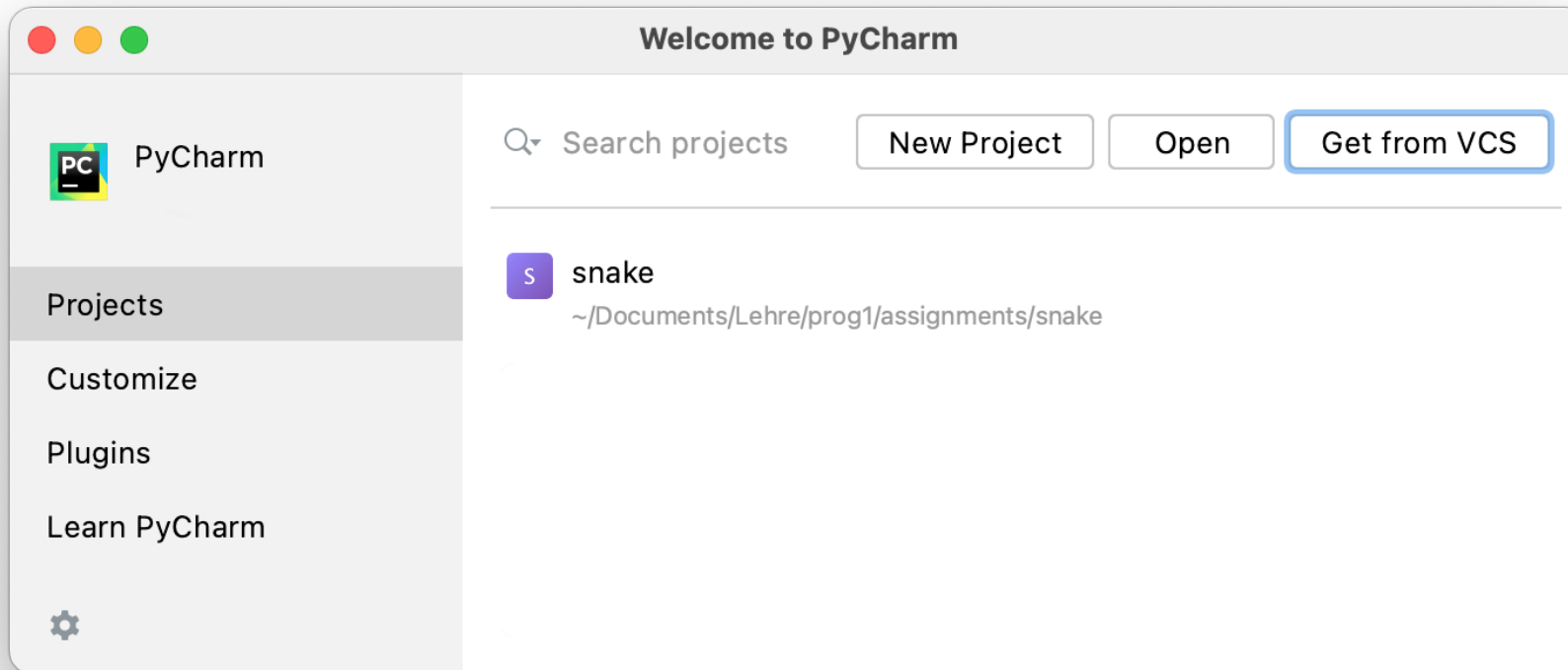
2 months ago

## Vorlage für eine Übungsaufgabe

 [Readme](#)

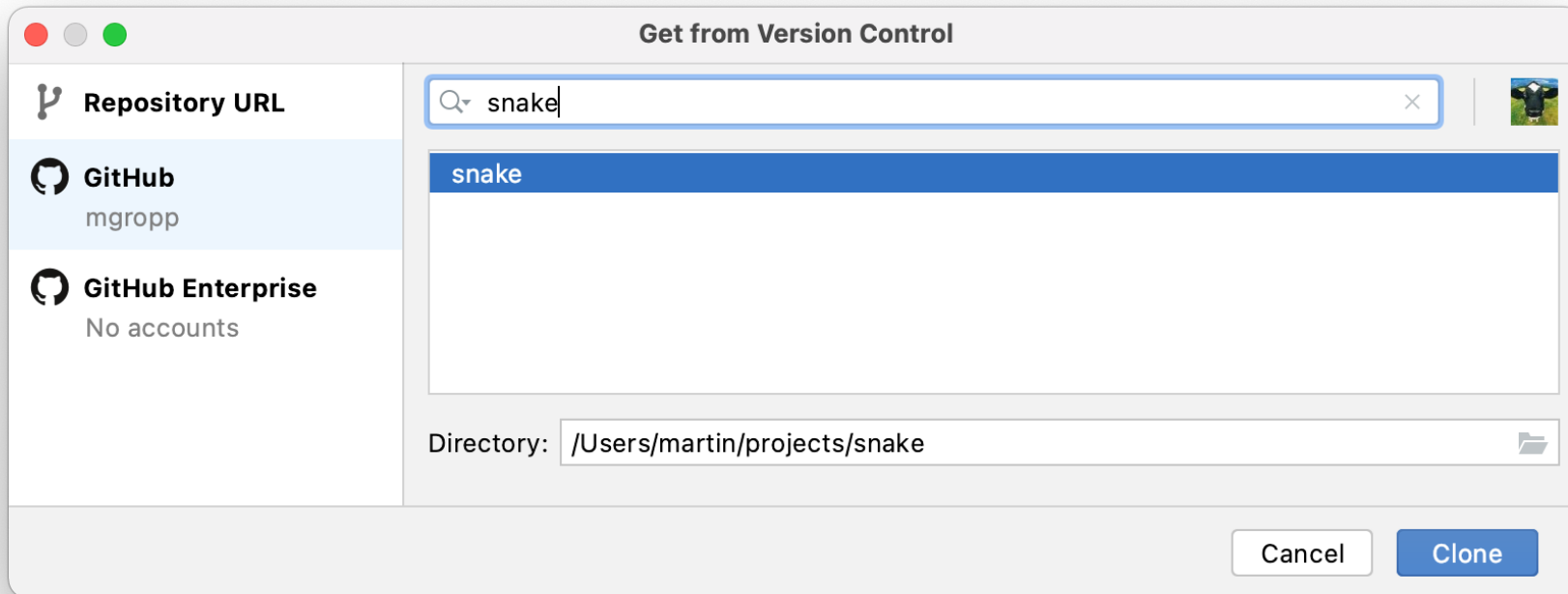
# PYCHARM: CLONE I

"Get from VCS" (VCS: version control system)



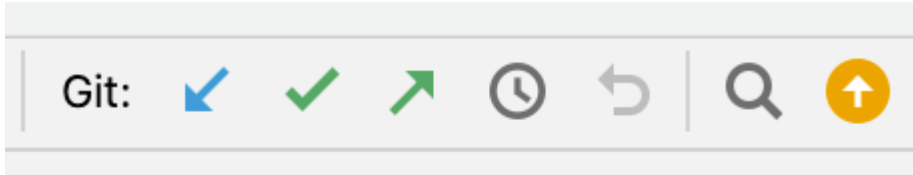


# PYCHARM: CLONE II (GITHUB)





# UPDATE, COMMIT, PUSH



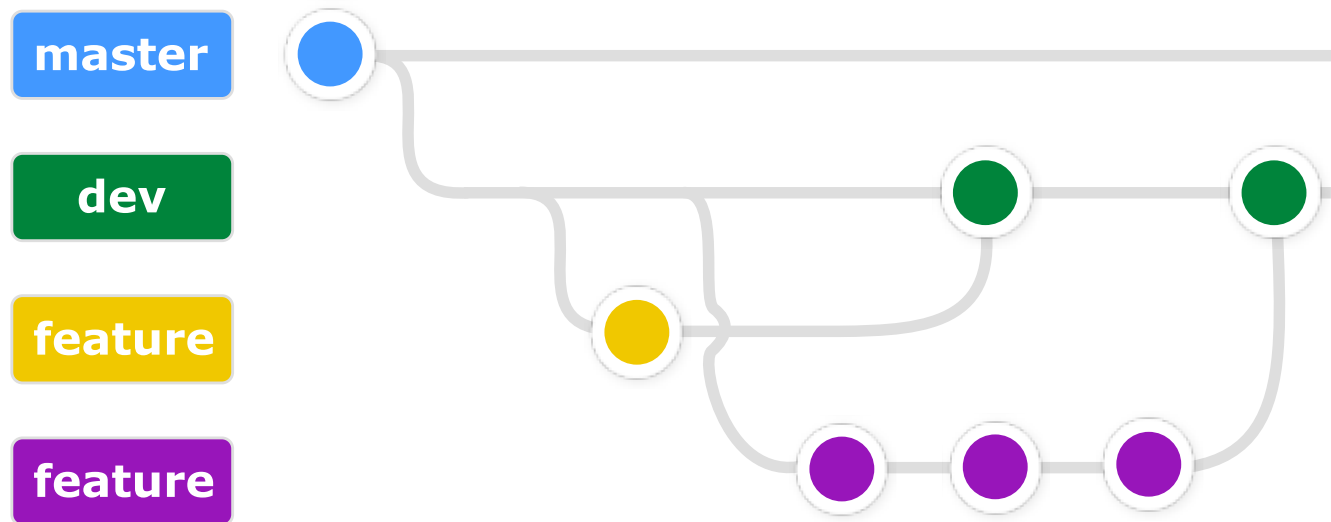
- **Update Project:** pull in changes from the remote repository
- **Commit:** record local changes
- **Push:** transfer local commits to the remote repository

Files need to be **added** to git before changes can be committed (right click → *git* → *Add*)

# ADVANCED TOPICS: BRANCHES



**Branches** are often used to work on a feature in isolation from the rest of the code, and then merge it back to a *master* or *main* branch when it is finished. When a developer has a branch *checked out* in their local repository, commits are made to this branch.



# ADVANCED TOPICS: COLLABORATION

To contribute to a software project, programmers will typically work on their changes in a new *branch* or even in their own *fork* of the repository.

They can then send a **merge request** / **pull request** to the maintainer of the project who will evaluate the changes and accept or reject them.

# ADVANCED TOPICS: CONFLICTS



**Conflicts:** When you collaborate with other people, try to avoid working on the same files at the same time. In many cases, git can automatically resolve resulting conflicts, but manual conflict resolution can be a lot of work.