LU04c - Working with GitHub Classroom
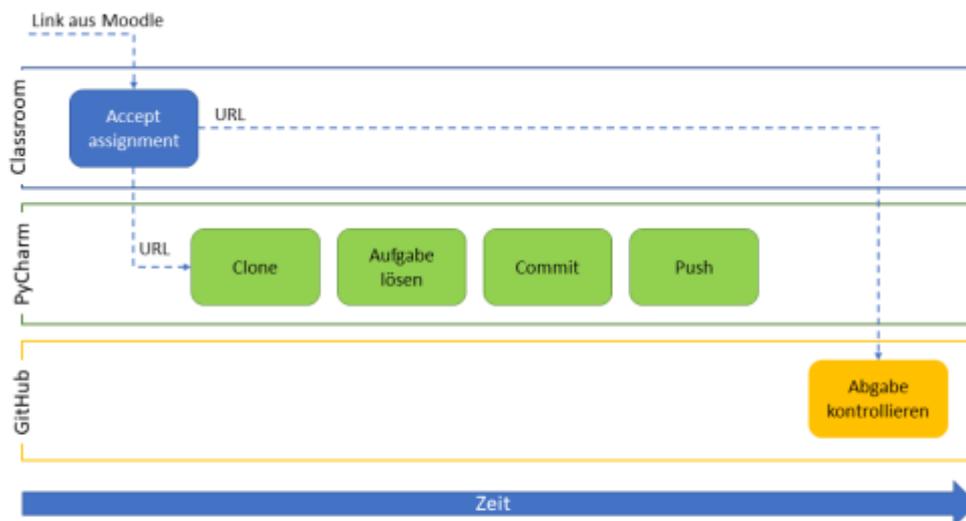
Video for working with GitHub, Classroom and PyCharm

Overview

For the programming tasks in module 319, we use GitHub Classroom and PyCharm (or a similar development environment). This allows you to learn the most important steps with git and you do not need to upload your solutions separately to Moodle.
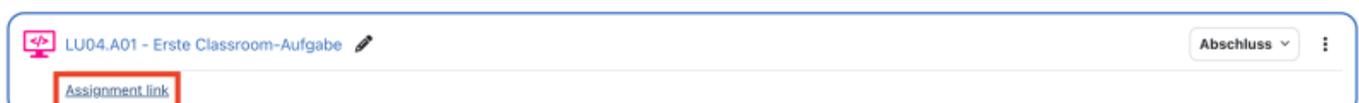

Prerequisite

Before you can work with GitHub Classroom, you must …

- … have created an account with GitHub.

Procedure

1. accept the assignment

In the Moodle course you will find the link to the assignment in the classroom.

For the first task in a new module:

Select your name from the list:

Join the classroom:

m319-ix24

To join the GitHub Classroom for this course, please select yourself from the list below to associate your GitHub account with your school's identifier (i.e., your name, ID, or email).

Can't find your name? Skip to the next step →

Identifiers	
IM24b – Lorin Esati	>
IM24b – Martin Rakitovszky	>
IM24b – Matteo Ceresetti	>
IM24b – Matteo Plessl	>
IM24b – Quentin Staquet	>
IM24b – Timo Nobs	>
IM24b – Yannick Tschurtschenthaler	>
Teacher – Andre Probst	>
Teacher – Christian Bolzern	>
Teacher – Kevin Maurizi	>

Open this link and accept the *assignment*.

m319–ix24

Accept the assignment —
m319-lu04-a01-classroom

Once you accept this assignment, you will be granted access to the `m319-lu04-a01-classroom-graphics80` repository in the m319-ix24 organization on GitHub.

Accept this assignment

Wait a few seconds and reload the website. You will see the address (URL) of your personal repository. Copy this address.

2. clone repository

Now open PyCharm and clone the repository as a new project using the copied link.
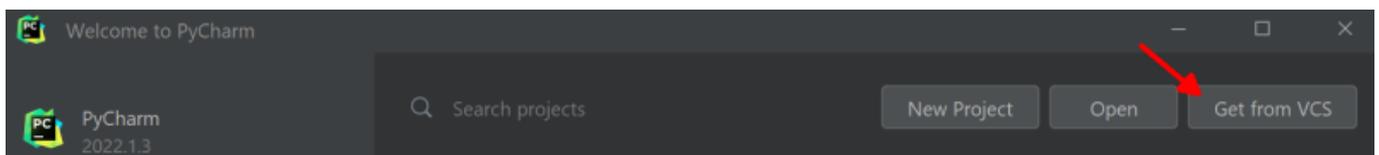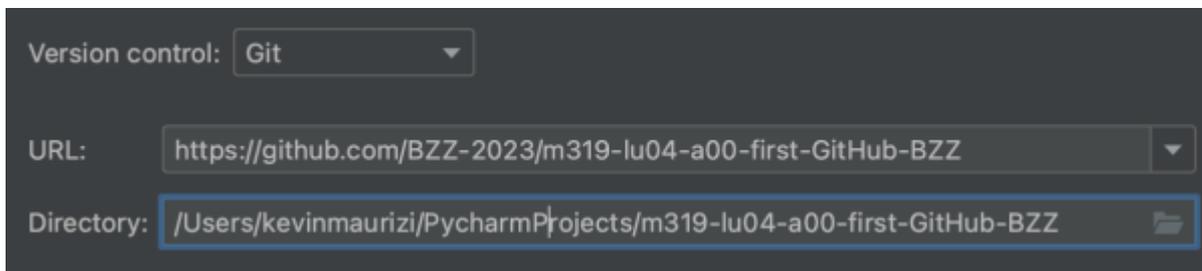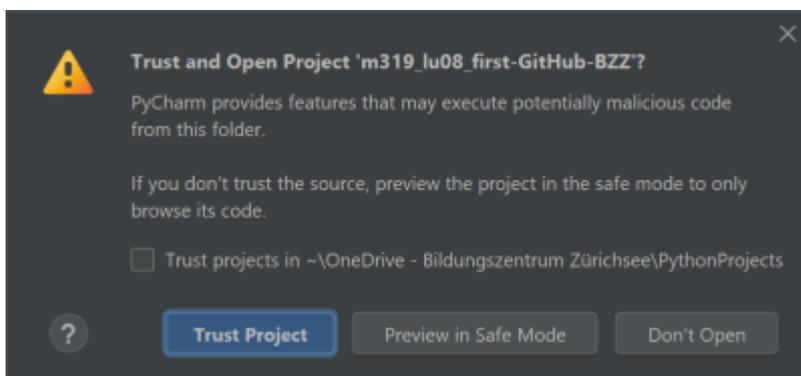

or
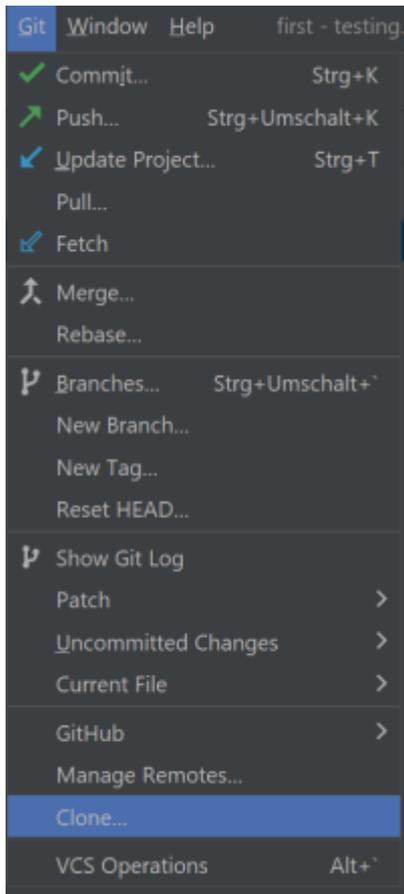
Enter the copied URL. If necessary, adjust the folder path in which the project is created.

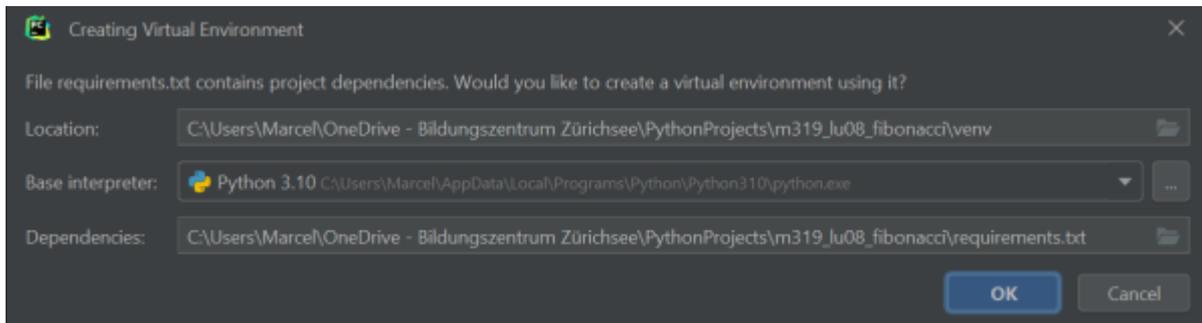You may then need to confirm that you trust this project.

3. set up the virtual environment

> The Python virtual environment contains all the external packages that our project requires. Pycharm can set this up automatically for us

Confirm the setup of the virtual environment with [OK]

4. solve the task

You now have a local copy of the project on your computer. The project contains several files and folders, only two of which are currently relevant to you:

- main.py: The Python script with the task.
- main_test.py: A Python script with which you can test your solution.

Search in the file `main.py` file, find the line with the comment `# TODO` …. Change the line according to the instructions in the comment. Then delete the comment.

5. test the solution

Automated testing of your solution initially has nothing to do with git or GitHub Classroom. It allows you to receive direct feedback on your solution.

Open the file `main_test.py` and execute the function `test()` function. If your solution is correct, you will see the following in the „run" window:

```
=========================== test session starts
===========================
collecting ... collected 1 item

main_test.py::test PASSED
[100%]


=========================== 1 passed in 0.01s
===========================
```

If there is still an error in your solution, the output will look something like this:
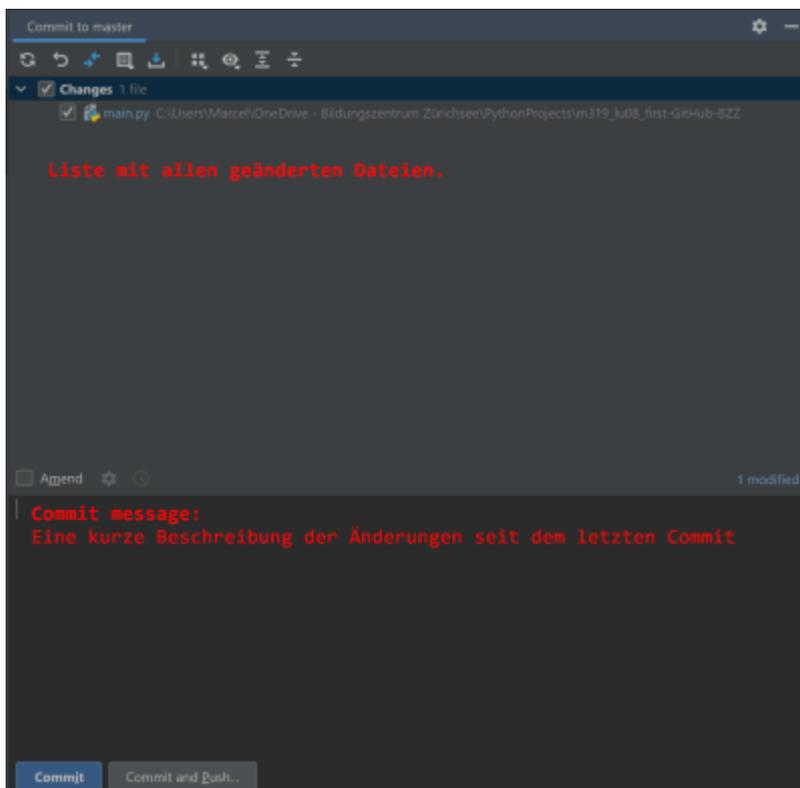
```
=========================== test session starts
```

```
==============================
collecting ... collected 1 item

main_test.py::test FAILED
[100%]
main_test.py:2 (test)
'Total: 47.25\n' != 'Total: 78.75\n'


Expected :'Total: 78.75\n'
Actual   :'Total: 47.25\n'
```

6. commit

Once you have completed the task, perform a commit. This commits the current status of your files.

1. in the menu, select **git** menu, select the **commit** from the menu. A new view opens:

2. in the **commit message** to indicate what you have changed.

3. click on [Commit]. You should now receive a confirmation that the commit was successful.

If you end the comment with # TODO … you will receive a warning. You can remove the comment and perform the commit again. Alternatively, you can also click on [Commit anyways].

7. push

To submit the completed task, perform a **push** of your repository. This uploads the changes from your

local repository to GitHub Classroom.

1. in the menu, select **git** menu, select the **push ...** from the menu. A new window opens:
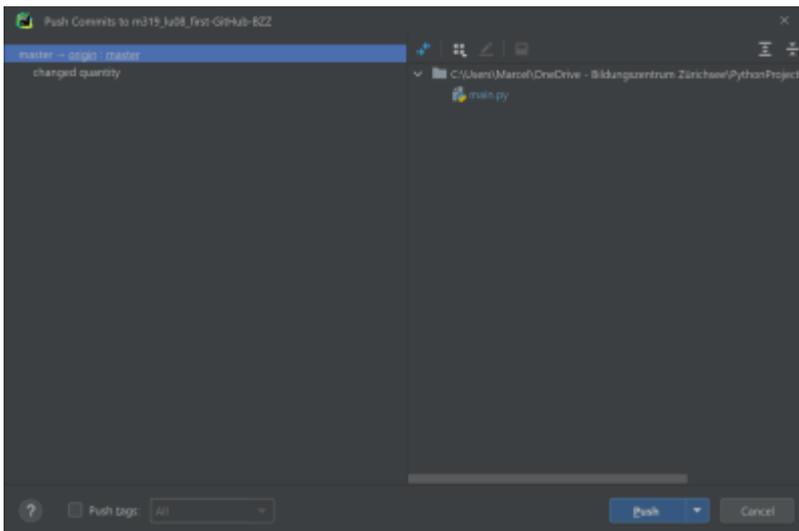
This window shows you which branch (e.g. master) is sent to which remote repository (e.g. origin). You will learn how to work with different branches later. For now, you can simply accept these settings.

2. click on [Push] and you will receive a confirmation that the push was successful.

Sometimes you will receive a message instead of the confirmation that you must first authorise yourself with GitHub.

In this case, select [Login via GitHub …]. This will open a new page in your web browser.

8. check submission

You can check your submission by viewing your repository in a web browser. To do this, open the link you used for **Clone project** you have created your local project.
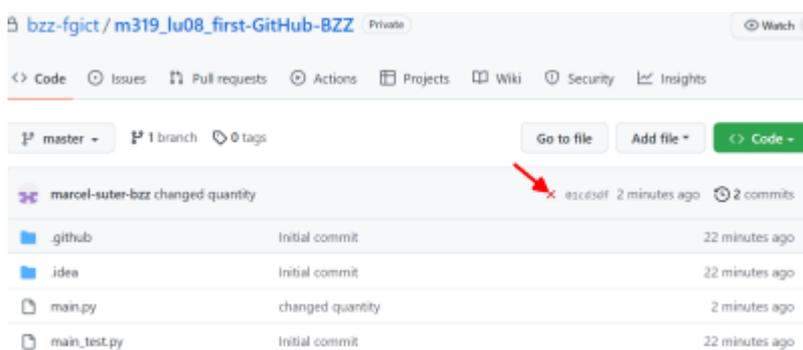
A small symbol shows you whether your solution is correct:

- Yellow circle: The check is still running. Wait 1-2 minutes and reload the page.
- Red cross: One or more tests have not been successful. Click on the red cross for more detailed information.
- Green tick: All tests were successful.

[m319-E1F](#)

From:

https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:

https://wiki.bzz.ch/en/modul/m319/learningunits/lu04/gitanwendung

Last update: **2025/06/23 07:45**