

GitHub desktop manual

Introduction to GitHub:

GitHub is a web-based platform revolutionizing collaborative software development through Git version control. It organizes projects into repositories, providing tools like pull requests and issues for seamless collaboration. GitHub Desktop, its user-friendly companion, offers a graphical interface simplifying Git tasks for both beginners and seasoned developers. With features like visual branch management, easy commits, and syncing capabilities, GitHub Desktop streamlines the process of working with repositories, enhancing the overall efficiency of version control. Together, GitHub and GitHub Desktop form a powerful combination, empowering teams to manage projects, track changes, and facilitate smooth collaboration in the world of software development.

Git Flow:

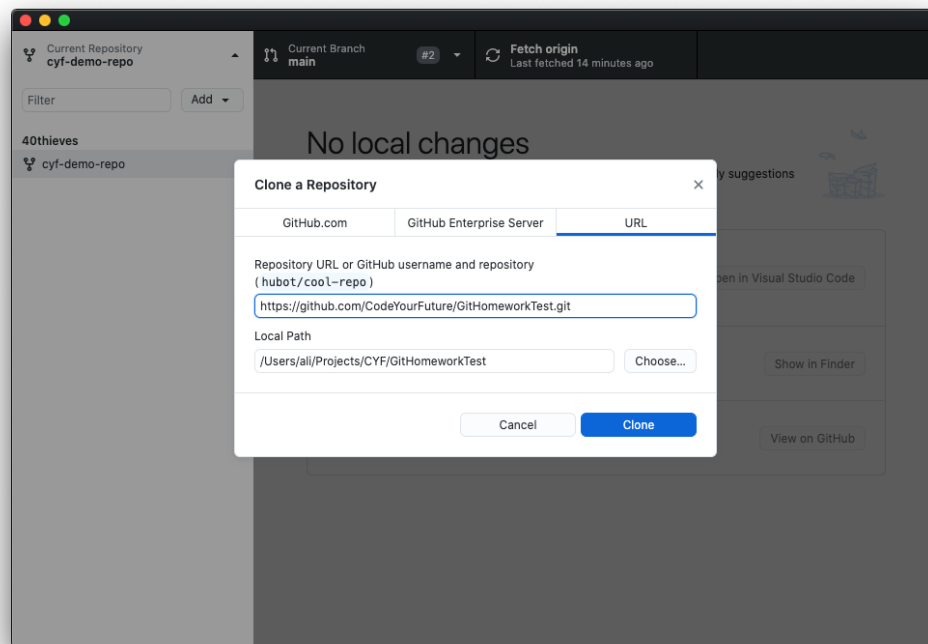
Git flow is a branching model and workflow for version control using Git, designed to streamline software development processes. It introduces a structured set of branches, including the “Master” or “Main” branch for production-ready code, the “develop” branch for ongoing development, feature branches for new features, release branches for preparing releases, and hotfix branches for urgent fixes. This model provides a systematic approach to collaboration, ensuring isolation of features during development, stability in the production code, and organized releases. Developers follow a set of conventions, creating branches for specific tasks and merging them back into the appropriate branches once completed. Git flow is widely adopted for its clarity, organization, and ability to manage complex development cycles effectively.

Creating an account:

You open GitHub.com sign up and sign in and then you download GitHub desktop and log into the same account there.

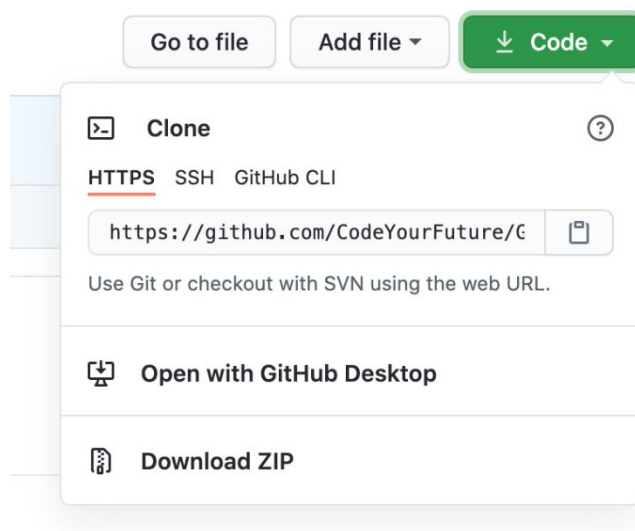
Cloning a repository:

1: after you open GitHub desktop click on file and clone repository and then URL. Should look like this:



2: to find the URL open github.com and log in

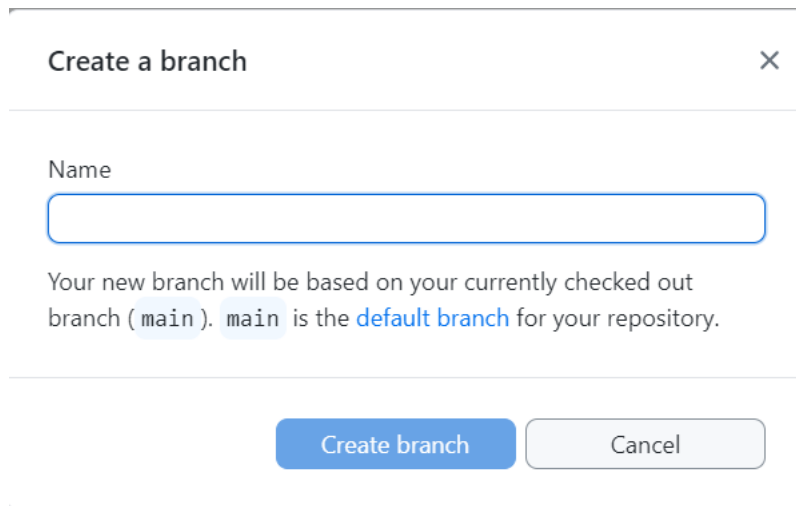
3: find the repository you are looking for and then click the green code button: (should look like this)



4: copy the URL and paste it in GitHub desktop pick a folder on your computer where you want the repository to go and click clone and it should be done.

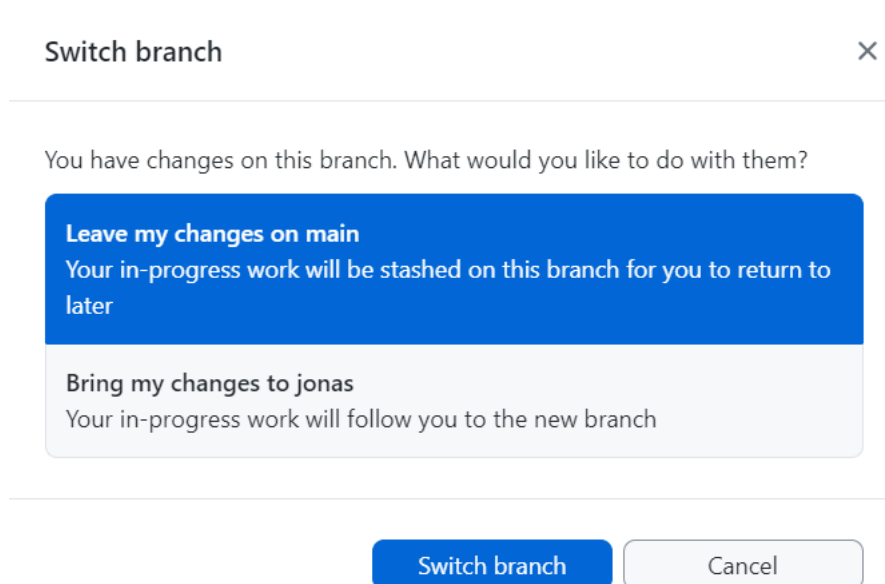
Creating a branch:

1: click on branch and new branch and then you can give your branch a name: (should look like this)



The screenshot shows a dialog box titled "Create a branch" with a close button (X) in the top right corner. Below the title bar, there is a label "Name" followed by an empty text input field. Below the input field, a message states: "Your new branch will be based on your currently checked out branch (main). main is the default branch for your repository." At the bottom of the dialog, there are two buttons: "Create branch" (highlighted in blue) and "Cancel" (in a light gray box).

2: then you will see this screen



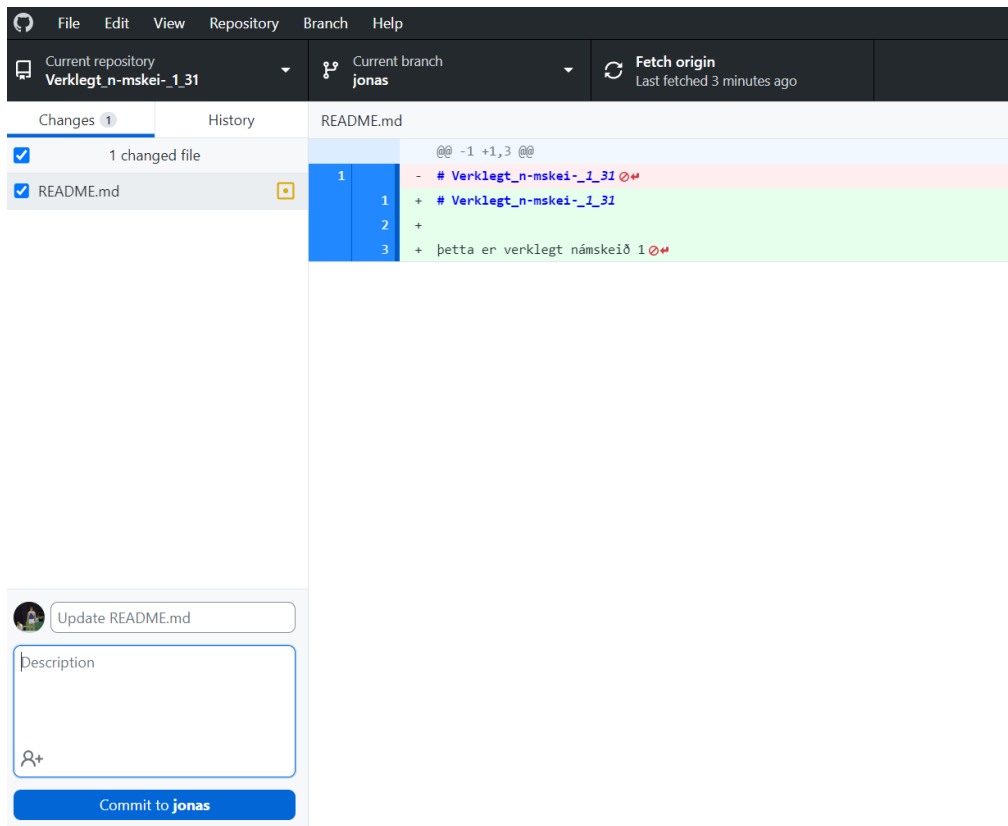
The screenshot shows a dialog box titled "Switch branch" with a close button (X) in the top right corner. Below the title bar, a message asks: "You have changes on this branch. What would you like to do with them?". There are two options presented in a list: "Leave my changes on main" (highlighted in blue) with the subtext "Your in-progress work will be stashed on this branch for you to return to later", and "Bring my changes to jonas" with the subtext "Your in-progress work will follow you to the new branch". At the bottom, there are two buttons: "Switch branch" (highlighted in blue) and "Cancel" (in a light gray box).

where you can choose either leave changes on main or bring them with to the new branch, then if you want to move to the new branch, you click switch branch otherwise cancel.

3: finally, if you want other people to be able to access the new branch you click the publish branch button.

Committing changes and pushing to current branch, pull requests and merge conflicts:

1: after making changes to the repository, you want to push to your current branch.(



2: on the right you can see the changes you made in red and green. To commit, you write your commit message where it says update README.md and then press commit to “your branch”.

3: then you press the “push origin” button.

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



Preview the Pull Request from your current branch

The current branch (jonas) is already published to GitHub. Preview the changes this pull request will have before proposing your changes.

Branch menu or `Ctrl + Alt + P`

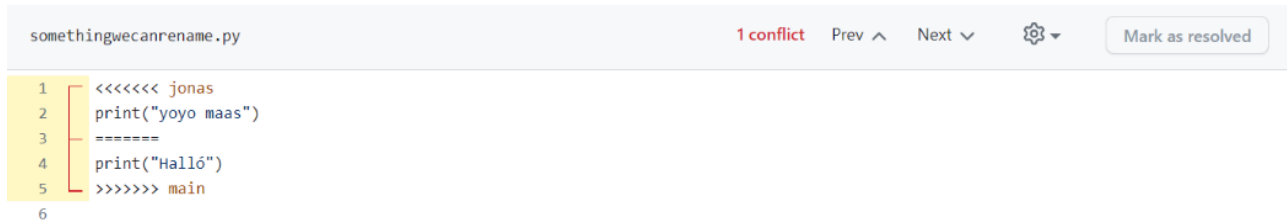
Preview Pull Request

4: to make the pull request you click on preview pull request and create pull request.

5: then you will be taken to GitHub.com where you press create pull request.

6: then they will tell you if there is a merge conflict to deal with, if there is no merge conflict you can merge the branches safely. If there is a merge conflict, we need to deal with it.

7: press “resolve conflict” then you should see a screen like this one:



The screenshot shows a code editor interface. At the top, the file name 'somethingwecanrename.py' is on the left, and '1 conflict' is on the right. To the right of '1 conflict' are buttons for 'Prev' (with an upward arrow), 'Next' (with a downward arrow), a settings gear icon, and a 'Mark as resolved' button. The code area shows a conflict between two versions of a file. A yellow highlight covers lines 1 through 5. On the left margin, a red bracket indicates the conflict range. The code content is as follows:

```
1 <<<<<< jonas
2 print("yoyo maas")
3 =====
4 print("Halló")
5 >>>>>> main
6
```

Where the yellow part is the merge conflict

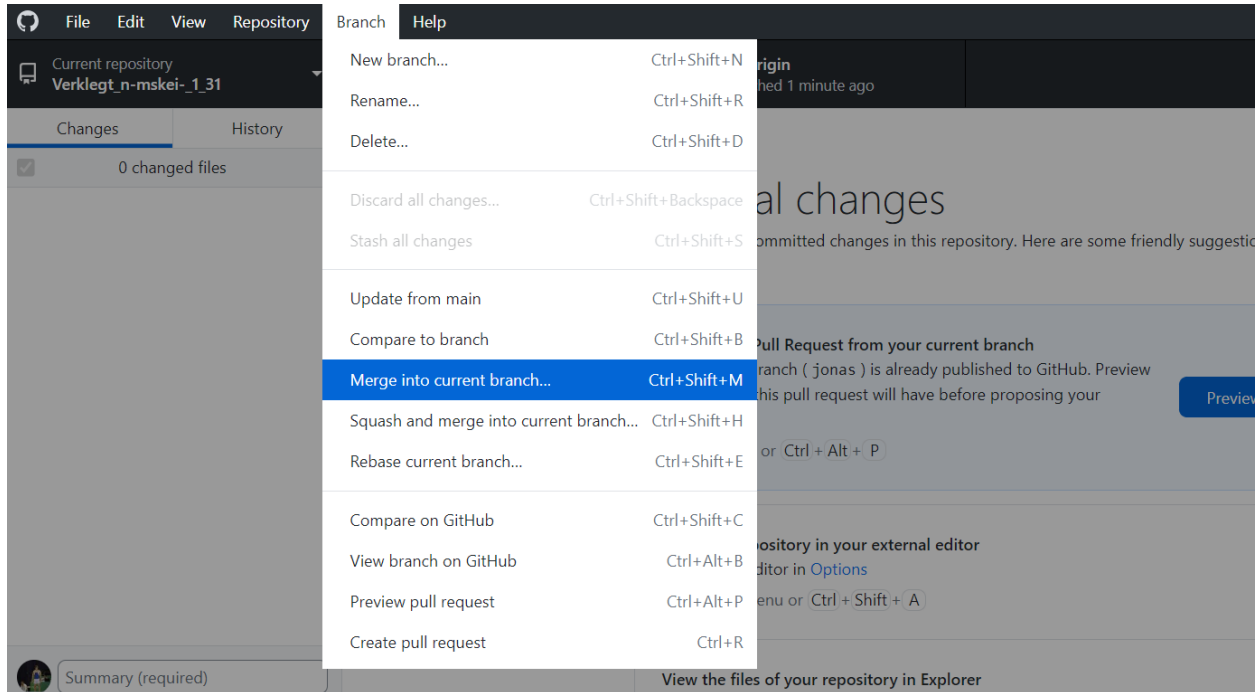
8: to fix it you must change all the yellow part to the code you want (which can be one or the other or even a mix of the two).

9: press mark as resolved and commit merge

Then go to GitHub desktop and fetch and pull.

Merging branches

After having pushed to your branch you can move to the branch you want to merge with. You click on “branch” and “merge into current branch”.



Then select the branch you pushed to and press “create merge commit”

Finally press “push origin”

Common problems:

1. **Cloning a repository:** Users may face issues cloning a repository. Ensure the correct repository URL, necessary permissions, and a stable internet connection.
2. **Push failures:** Sync or push operations may fail. Check for uncommitted changes, verify write access, and ensure remote repository accessibility.
3. **Branching issues:** Users might encounter problems with branches not appearing or syncing correctly. Ensure you're on the correct branch, sync to fetch branches, and check for branch creation permissions.
4. **Authentication problems:** Users may experience repeated authentication prompts. Verify GitHub credentials in settings, ensure account status, and use personal access tokens for two-factor authentication.
5. **Pushing to a protected branch:** Pushing to a protected branch may be restricted. Confirm permissions and ensure that required checks like status checks and code review approvals pass.