```
r[[rg_mm$max]] <- terra::extend(r[[rg_mm$max]], ext.combi)

# shift the smaller one over to the other side
ext.min <- terra::ext(r[[rg_mm$min]])
ext.min <- terra::ext(as.numeric(cc.xmin - min(rg)), cc.xmin, ext.min[3], ext.min[4])

terra::ext(r[[rg_mm$min]]) <- ext.min

# extent the smaller one too, resample to larger one
r[[rg_mm$min]] <- terra::extend(r[[rg_mm$min]], ext.combi)
r[[rg_mm$min]] <- terra::resample(r[[rg_mm$min]], r[[rg_mm$max]])

# fuse rasters over grid end
r <- terra::merge(r[[1]], r[[2]])
```

EAGLE M.Sc., MB2 2023/24

Introduction into programming using R

Part II: Concepts, code & practical exercises

Agenda

- Recap on git
- Exploring *poll.md* based on what you have learned today in the morning

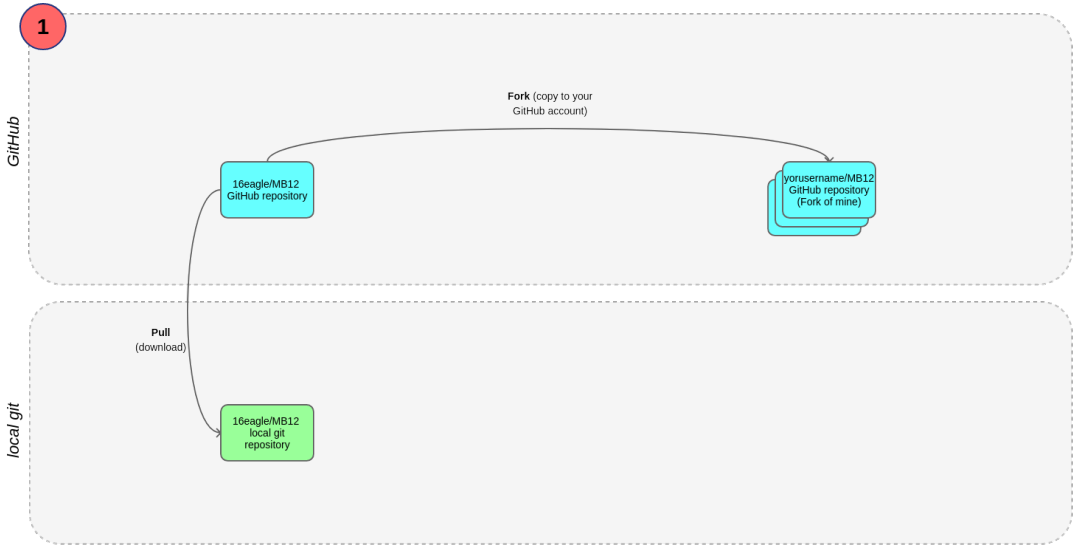
What did we use git and GitHub for last time?
Which steps did we apply?

Short recap

You have learned how to work with

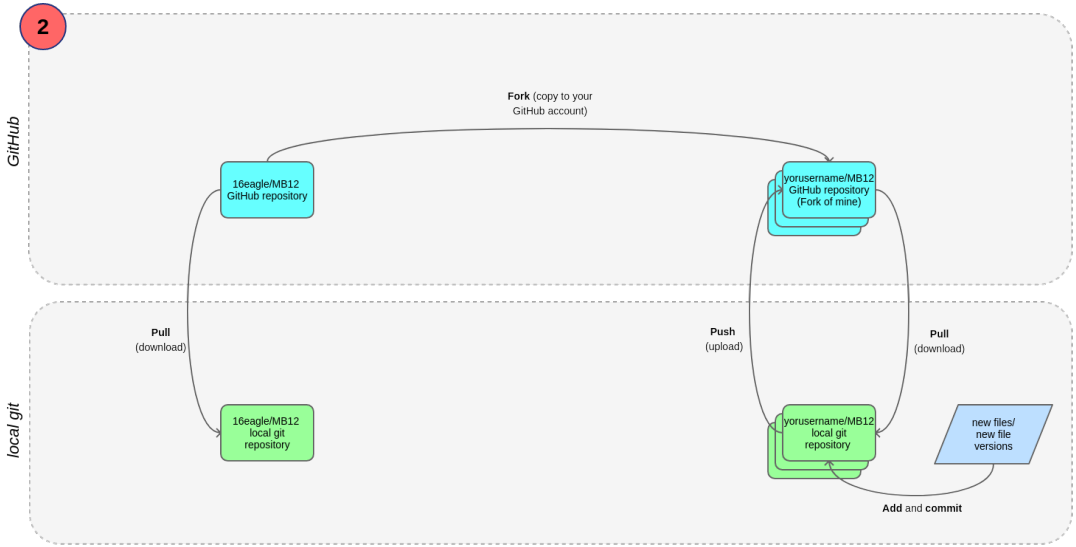
- git for local version control
- git and GitHub for publishing repos
- git and GitHub for collaboration (PR/merging)

Git workflow



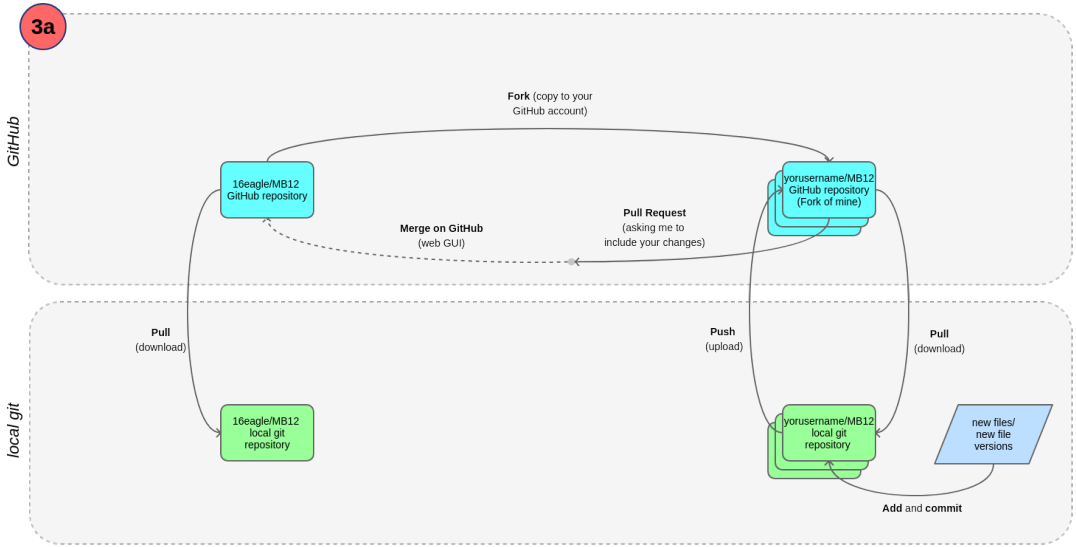
1 Schematics of our GitHub and git workflow

Git workflow



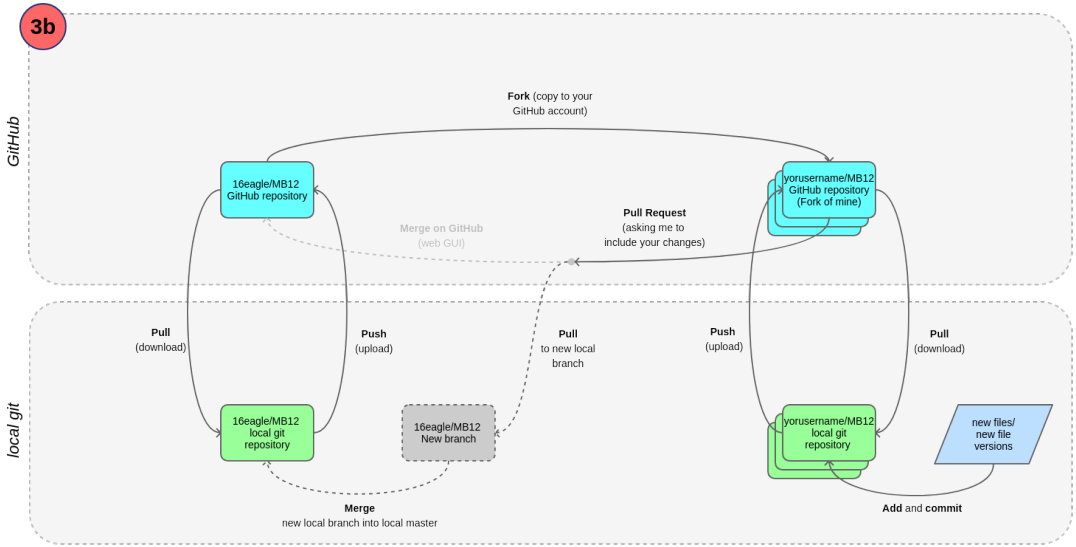
2 Schematics of our GitHub and git workflow

Git workflow



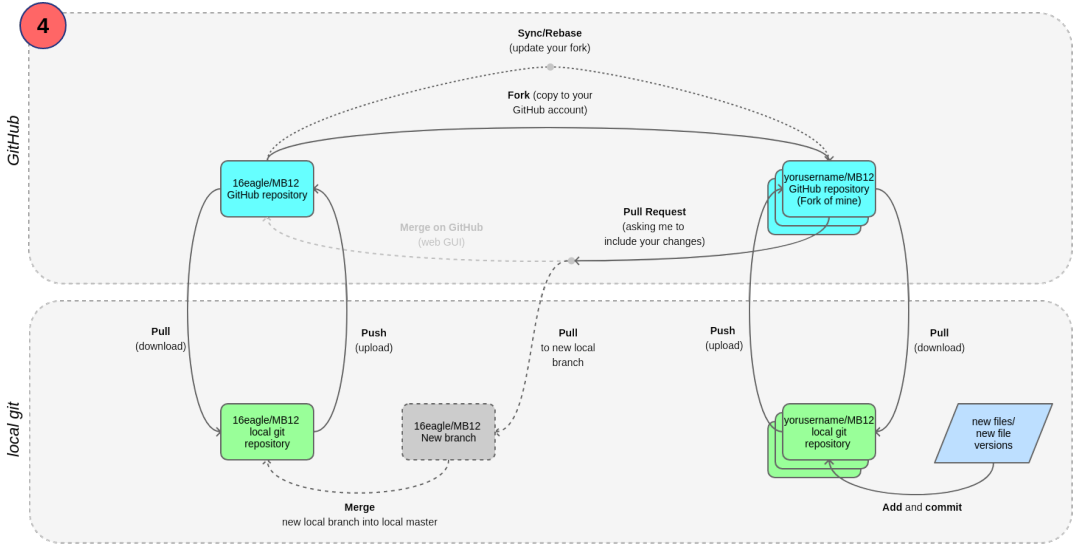
3 Schematics of our GitHub and git workflow

Git workflow



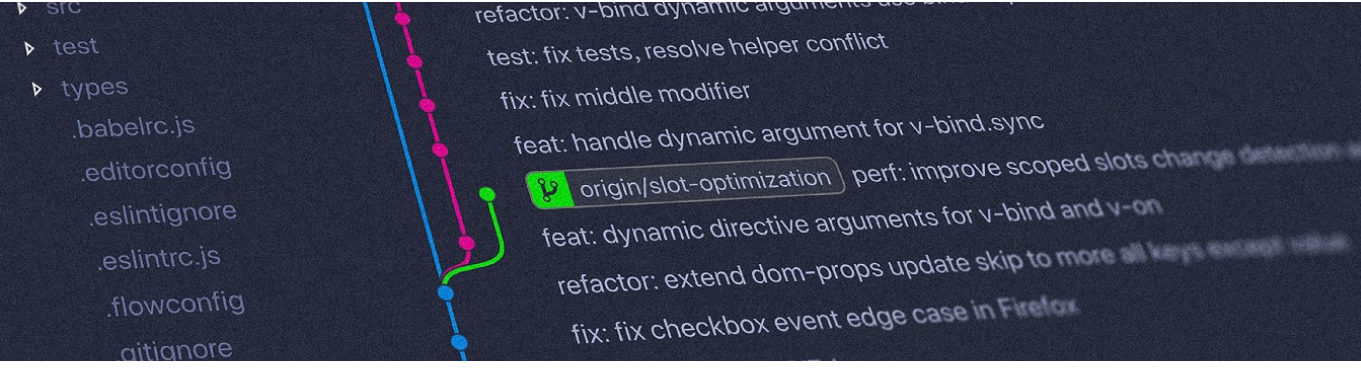
4 Schematics of our GitHub and git workflow

Git workflow



5 Schematics of our GitHub and git workflow

Any questions regarding last week's topic?



Follow-up:

Version control and collaboration using git II

Recap: Pulling your fork using the git cli

In your local directory that you want to use for the project (use *cd* to get there):

```
git init
git remote add origin https://github.com/yourGitHubUser/MB12.git
git pull origin master
```

Make your changes (edit the markdown file, copy new files etc.), then:

```
git add myfile.R
git commit -m "my commit message"
git push -u origin master
```

Now, you may create a *Pull Request* at `https://github.com/16EAGLE/MB12`. Next time, you will se how to merge a PR on the command line instead of using github's web GUI.

Git: Merging your Pull Request using the git cli

After you submitted your PR, I can either merge it on github.com (web GUI) or I can do that from the command line: In my local project directory, I create a new branch that contains the Pull Request:

```
git checkout -b someone-master master
git config pull.rebase false
git pull https://github.com/someone/MB12.git master
```

If there are conflicts, they need to be resolved now by editing the conflicting files and commiting the changes, e.g. in RStudio. Then, we can merge into master and push upstream:

```
git checkout master
git merge --no-ff someone-master
git push origin master
```

Done! The PR should now be merged and closed.

Updating your forked repositories

Syncing and rebasing

Git: Syncing, rebasing

After I have merged your changes into my MB12 repository (as just shown), *your* forks are now outdated (i.e. do not include the most recent changes in my repository).

Two ways of "updating" your fork:

- *Syncing* your fork in GitHub (follow `https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/syncing-a-fork`
- *Rebasing* your local repository based on mine, e.g. from the command line (see next slide)

Git: Rebasing your local git

Make sure your terminal is inside the directory of your local MB12 git repo!
Add my git repo as an upstream repo:

```
1        git remote add upstream https://github.com/16EAGLE/MB12.git
2        git fetch upstream
```

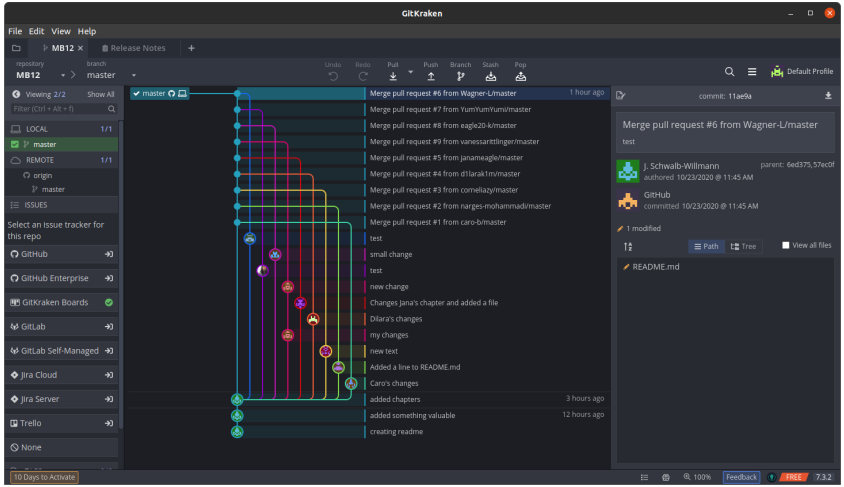
Make sure you are on master, then rebase:

```
1        git checkout master
2        git rebase upstream/master
```

Now, you can pull everything so that your repo matches mine again:

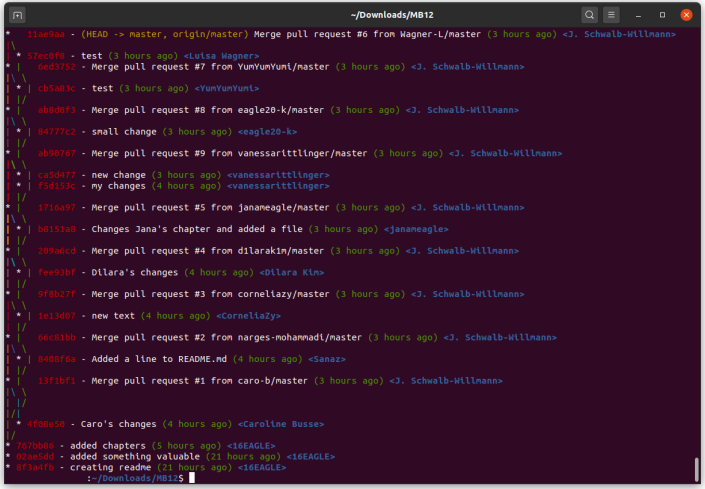
```
1        git pull -f origin master
```

Follow up on git: Git GUIs and visualizer



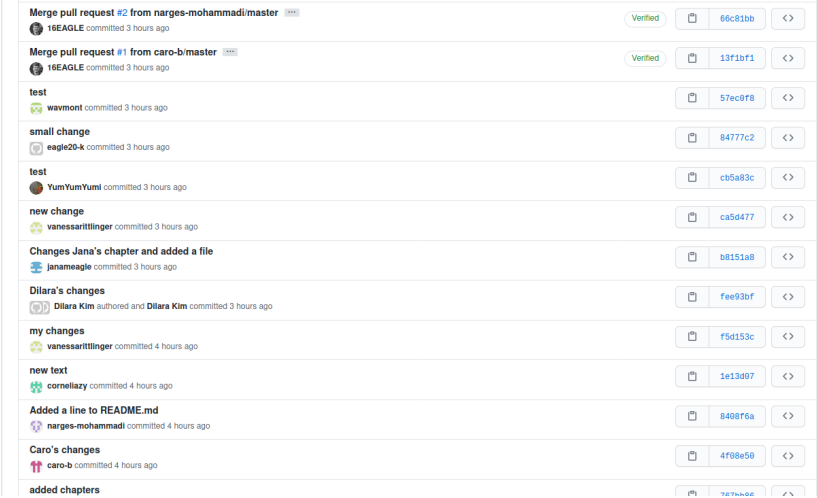
6 Gitkraken for Linux – *screenshoted*

Follow up on git: Git GUIs and visualizer



7 The git command itself – *screenshoted*

Follow up on git: Git GUIs and visualizer



8 GitHub – *screenshoted*

Follow up on git: Git GUIs and visualizer

Two further sources:

- On Linux: https://git-scm.com/download/gui/linux
- General: https://livablesoftware.com/tools-to-visualize-the-history-of-a-git-repository/

Live coding

Exploring *poll.md* using R

Any questions?

Feedback

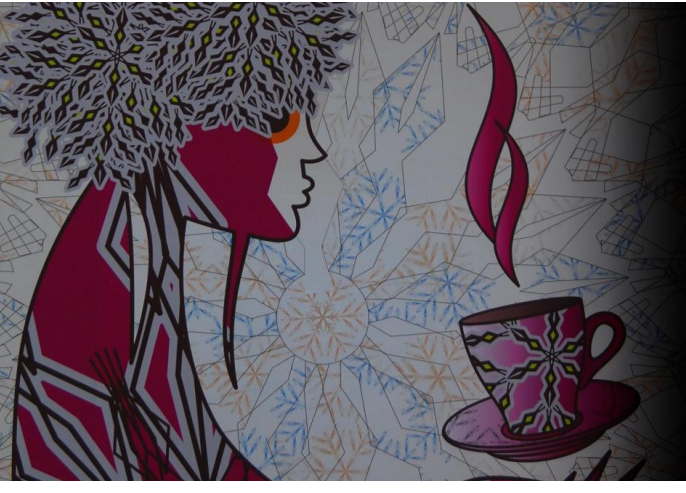
Anonymous feedback via slido.com, event code **#mb2training**.

What we would like to know from you:

- Is the speed ok?
- Is the session structure ok?
- Is there any content you'd like to be covered?
- Is it boring or not?
- Is it helpful or not?
- Is the presentation style ok?
- Anything else?

Go to slido.com & join #mb2training

Please give feedback using the feedback box.



Coffee time – see you next week