

SFB 1258

Neutrinos
Dark Matter
Messengers



git tutorial

Andi Mathis

Technische Universität München

15.06.2018

THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



<https://xkcd.com/1597/>

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT,

Today's goal: No more need to memorize – all commands contained in these slides

NB: the approach works nevertheless quite well...



<https://xkcd.com/1597/>

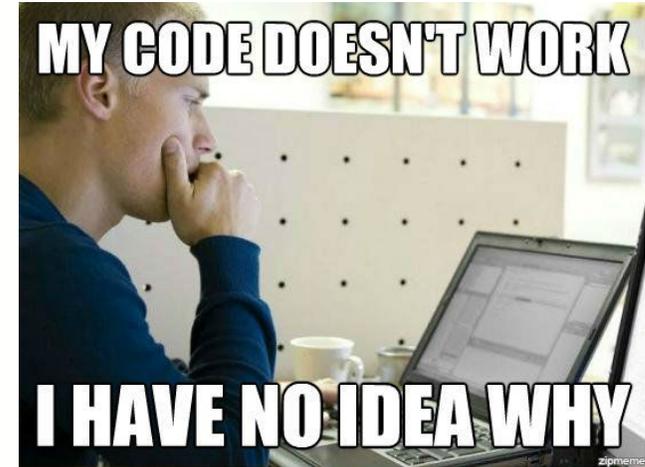
Why bothering?

- Two simple questions:
- What changes did you do to your code yesterday
- Are you ad hoc able to reproduce your code as of the day before?
- Me: "I can barely remember yesterday..."

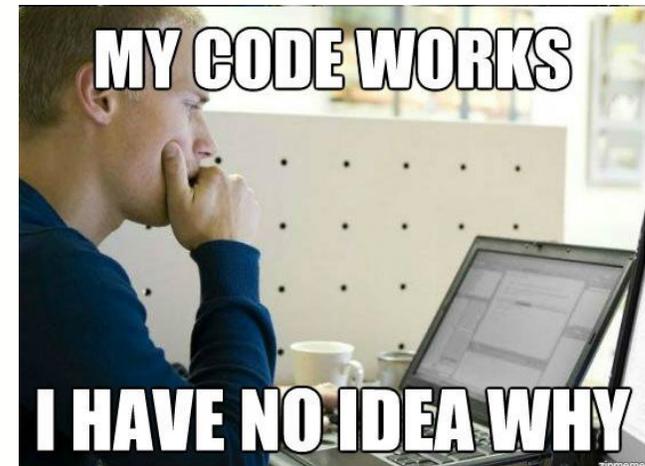
- In terms of debugging both are equally bad...
 - Knowing the most recent changes is helping a lot!

- Plus: what do you do when developing code together?
 - Shared folder in dropbox? Enjoy...

- Help? See e.g. <https://try.github.io>
 - ... from which some material is used in this presentation



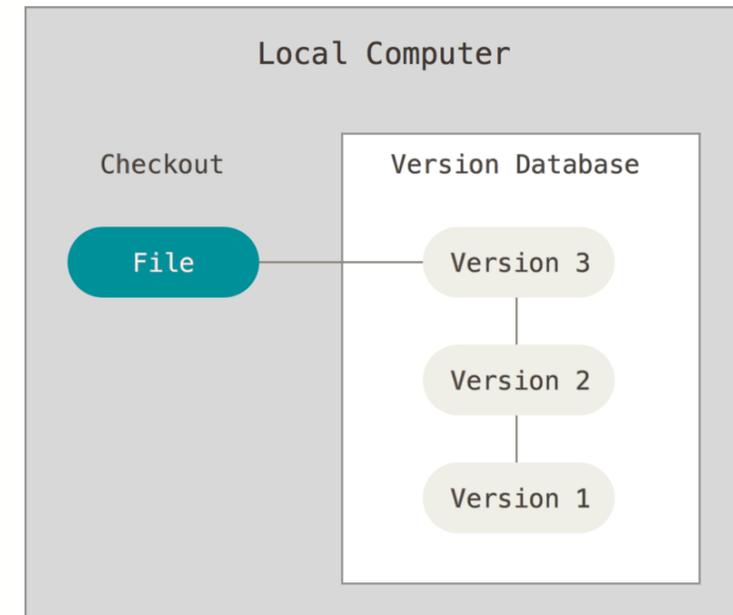
VS.



Let's start easy – simple versioning with git

Simple local versioning of files

- `git init` in the folder you want to have under version control*
- Adds `.git` directory which contains **all** the history, configuration, ...
- `git status` to see the status of the repository
 - On branch `master` (let's discuss branches a bit later)
 - No commits yet
 - nothing to commit (create/copy files and use "git add" to track)
- What is a commit?
 - A snapshot of all the files in your directory
 - (of course git does not always save ALL the files...)



* In case you never used git before it might be that you have to provide your name etc. – git will ask you about that

Let's try!

- Let's imagine we start writing a paper - set up a skeleton
- `git status` to see the status of the repository
 - On branch master
 - No commits yet
 - Untracked files:
 - (use "`git add <file>...`" to include in what will be committed)
 - `paper.tex`
- Cool, it even tells us what to do!
 - We want to keep `paper.tex` tracked
 - `git add paper.tex`
- That's it???

```
User@computer MINGW64 /d/Desktop
$ git status
It's complicated

User@computer MINGW64 /d/Desktop
$
```

When you mess up your
version control

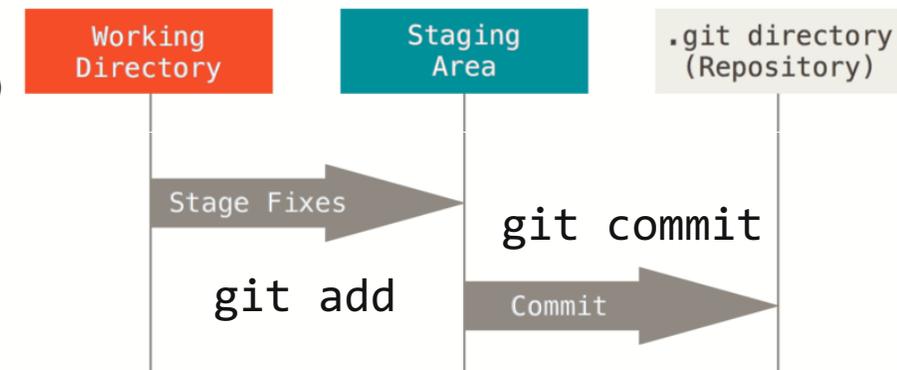
And that's it?

- No, we still need to commit!
- `git commit`
- And then enter a nice, **meaningful** commit message
 - Unhappy with the editor?
 - `git config --global core.editor " <editor of your choice> "`
- Let's see what we did – take a look at the history: `git log`
- `commit 8884f9b3244a9627404de30da96a72ec0cd1df4b (HEAD -> master)`
- Author: Andreas Mathis <andreas.mathis@cern.ch>
- Date: Wed Jun 13 14:17:30 2018 +0200
- Add skeleton for the paper
- The commit hash is a unique identified of a commit!

	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAAANDS	2 HOURS AGO

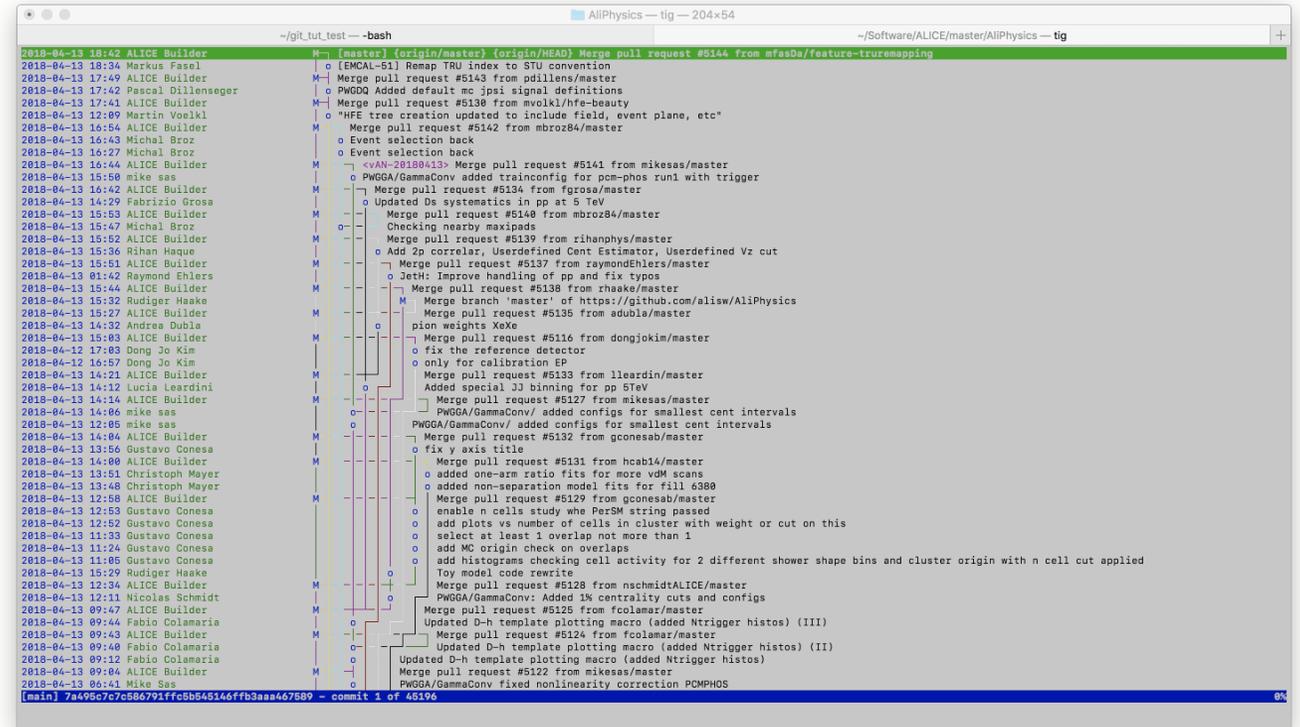
AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

<https://xkcd.com/1296/>



How to obtain more information?

- Take a look who did changes to the most recent version of the file?
- `git blame <filename>`
- There are a few tools to access commit information
 - My preference: `tig`
 - Mac: `brew install tig`
 - Ubuntu: `apt-get install tig`



```
~/Software/ALICE/master/AllPhysics -- tig
2018-04-13 18:42 ALICE Builder M [main] (origin/master) (origin/HEAD) Merge pull request #5144 from mfasDa/feature-trueremapping
2018-04-13 18:34 Markus Fasel M [EMCAL-51] Remap TRU index to STU convention
2018-04-13 17:49 ALICE Builder M Merge pull request #5143 from pdillens/master
2018-04-13 17:42 Pascal Dillenseger M PWG/QQ Added default mc jpsi signal definitions
2018-04-13 17:41 ALICE Builder M Merge pull request #5138 from mvolk1/hfe-beauty
2018-04-13 12:09 Martin Voeikl M "HFE tree creation updated to include field, event plane, etc"
2018-04-13 16:54 ALICE Builder M Merge pull request #5142 from mbroz84/master
2018-04-13 16:43 Michal Broz M Event selection back
2018-04-13 16:27 Michal Broz M Event selection back
2018-04-13 16:44 ALICE Builder M Merge pull request #5141 from mikesas/master
2018-04-13 15:58 mike sas M PWGGA/GammaConv added trainconfig for pcm-phos run1 with trigger
2018-04-13 16:42 ALICE Builder M Merge pull request #5134 from fgrasa/master
2018-04-13 14:29 Fabrizio Grossa M Updated Ds systematics in pp at 5 TeV
2018-04-13 15:53 ALICE Builder M Merge pull request #5140 from mbroz84/master
2018-04-13 15:47 Michal Broz M Checking nearby maxipads
2018-04-13 15:52 ALICE Builder M Merge pull request #5139 from rihanphys/master
2018-04-13 15:36 Rihan Haque M Add 2p correlator, Userdefined Cent Estimator, Userdefined Vz cut
2018-04-13 15:51 ALICE Builder M Merge pull request #5137 from raymondEhlers/master
2018-04-13 01:42 Raymond Ehlers M JetH: Improve handling of pp and fix typos
2018-04-13 15:44 ALICE Builder M Merge pull request #5138 from shakeo/master
2018-04-13 15:32 Rudiger Haake M Merge branch 'master' of https://github.com/alice/AllPhysics
2018-04-13 15:27 ALICE Builder M Merge pull request #5135 from adubla/master
2018-04-13 14:32 Andrea Dubla M pion weights XxXe
2018-04-13 15:03 ALICE Builder M Merge pull request #5116 from dongjokim/master
2018-04-12 17:03 Dong Jo Kim M fix the reference detector
2018-04-12 16:57 Dong Jo Kim M only for calibration EP
2018-04-13 14:21 ALICE Builder M Merge pull request #5133 from llearnin/master
2018-04-13 14:12 Lucia Leardini M Added special JJ binning for pp 5TeV
2018-04-13 14:14 ALICE Builder M Merge pull request #5127 from mikesas/master
2018-04-13 14:06 mike sas M PWGGA/GammaConv/ added configs for smallest cent intervals
2018-04-13 12:05 mike sas M PWGGA/GammaConv/ added configs for smallest cent intervals
2018-04-13 14:04 ALICE Builder M Merge pull request #5132 from gconesab/master
2018-04-13 13:56 Gustavo Conesa M fix y axis title
2018-04-13 14:00 ALICE Builder M Merge pull request #5131 from hcab14/master
2018-04-13 13:51 Christoph Mayer M added one-arm ratio fits for more v0M scans
2018-04-13 13:40 Christoph Mayer M added non-separation model fits for fill 6380
2018-04-13 12:58 ALICE Builder M Merge pull request #5129 from gconesab/master
2018-04-13 12:53 Gustavo Conesa M enable n cells study when Par3M string passed
2018-04-13 12:52 Gustavo Conesa M add plots vs number of cells in cluster with weight or cut on this
2018-04-13 11:33 Gustavo Conesa M select at least 1 overlap not more than 1
2018-04-13 11:24 Gustavo Conesa M add MC origin check on overlaps
2018-04-13 11:05 Gustavo Conesa M add histograms checking cell activity for 2 different shower shape bins and cluster origin with n cell cut applied
2018-04-13 15:29 Rudiger Haake M Toy model code rewrite
2018-04-13 12:34 ALICE Builder M Merge pull request #5128 from nschmidt/ALICE/master
2018-04-13 12:11 Nicolas Schmidt M PWGGA/GammaConv: Added 1% centrality cuts and configs
2018-04-13 09:47 ALICE Builder M Merge pull request #5126 from fcolamar/master
2018-04-13 09:44 Fabio Colamar M Updated D-h template plotting macro (added Ntrigger histos) (III)
2018-04-13 09:43 ALICE Builder M Merge pull request #5124 from fcolamar/master
2018-04-13 09:40 Fabio Colamar M Updated D-h template plotting macro (added Ntrigger histos) (II)
2018-04-13 09:12 Fabio Colamar M Updated D-h template plotting macro (added Ntrigger histos)
2018-04-13 09:04 ALICE Builder M Merge pull request #5122 from mikesas/master
2018-04-13 06:41 Mike Sas M PWGGA/GammaConv fixed nonlinearity correction PCMPHOS
[main] 7a4967c7c586791ff65b545146ffb3aaa467589 - commit 1 of 45196
```

Differences between the versions

- Before committing – what changed?
 - To see changes in your working directory prior to a commit: `git diff`
- Reverting changes from the staging area - **DANGER ZONE!**
 - The file is already committed: `git checkout <file>`
 - The file is new: `git reset HEAD <file>`
 - Reset all changes w.r.t. the last commit: `git reset --hard HEAD`
- Commit the changes
 - Save some time:
 - `git commit -m "commit message"`
- Oops, I forgot to add a file to my commit!
 - `git add <file>`
 - `git commit --amend`

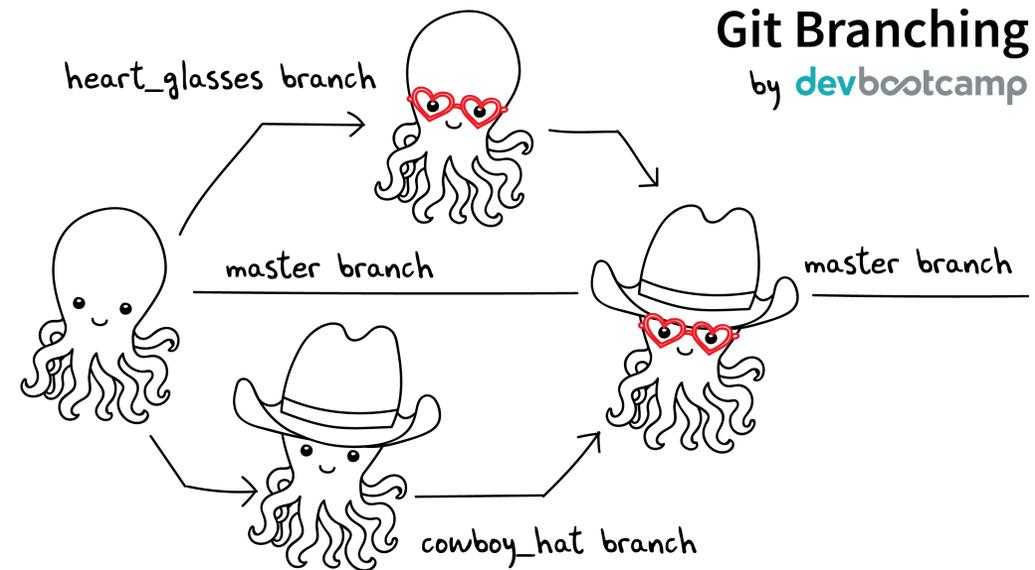
Ignoring files

- Let's compile our tex project!
- git status
 - On branch master
 - Untracked files:
 - (use "git add <file>..." to include in what will be committed)
 - paper.aux
 - paper.log
 - paper.pdf
- Can become annoying when files pile up....
 - Remedy: **.gitignore**
 - Add files you do not want to have under version control to an empty text file named **.gitignore**
 - Trick: Also wild cards work, e.g. ***.pdf**
 - Add & commit it

Branching

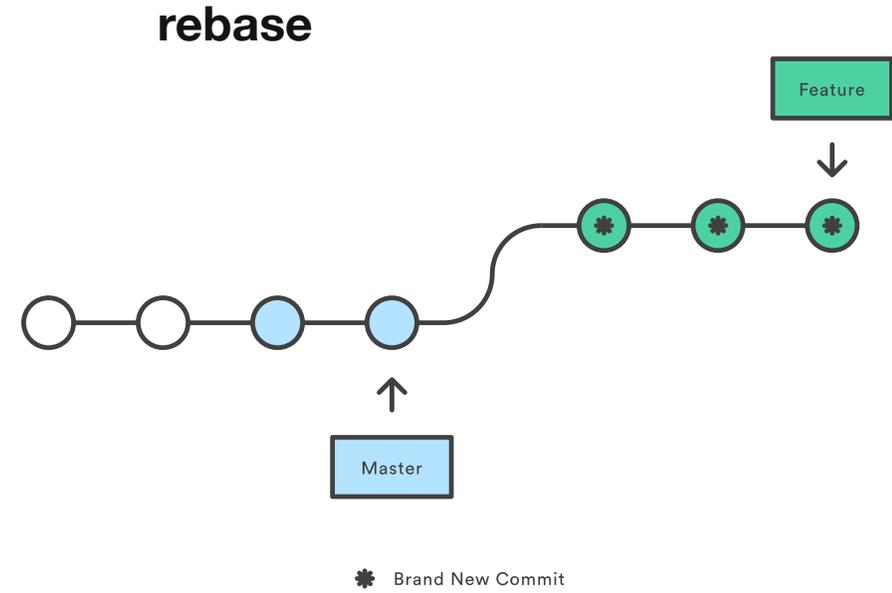
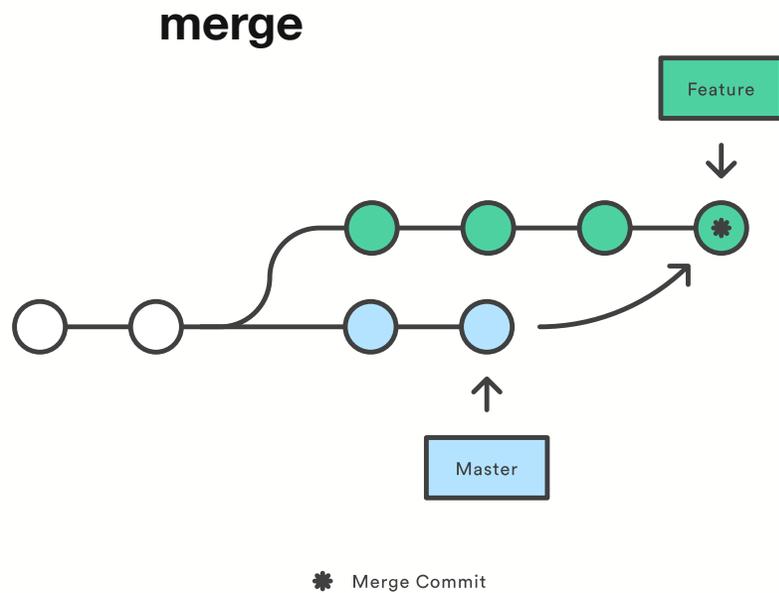
When is branching helpful?

- Imagine you cannot decide to which journal you want to submit to
 - Create different versions of the paper, but keep e.g. the intro the same
- When developing different, independent features at the same time
 - Keep a working master!
- What branches do we have right now?
 - `git branch`
- Create a new one
 - `git branch <new branch name>`
- Switch to the branch
 - `git checkout <new branch name>`
 - (do both creation & switch at the same time: `git checkout -b <new branch name>`)



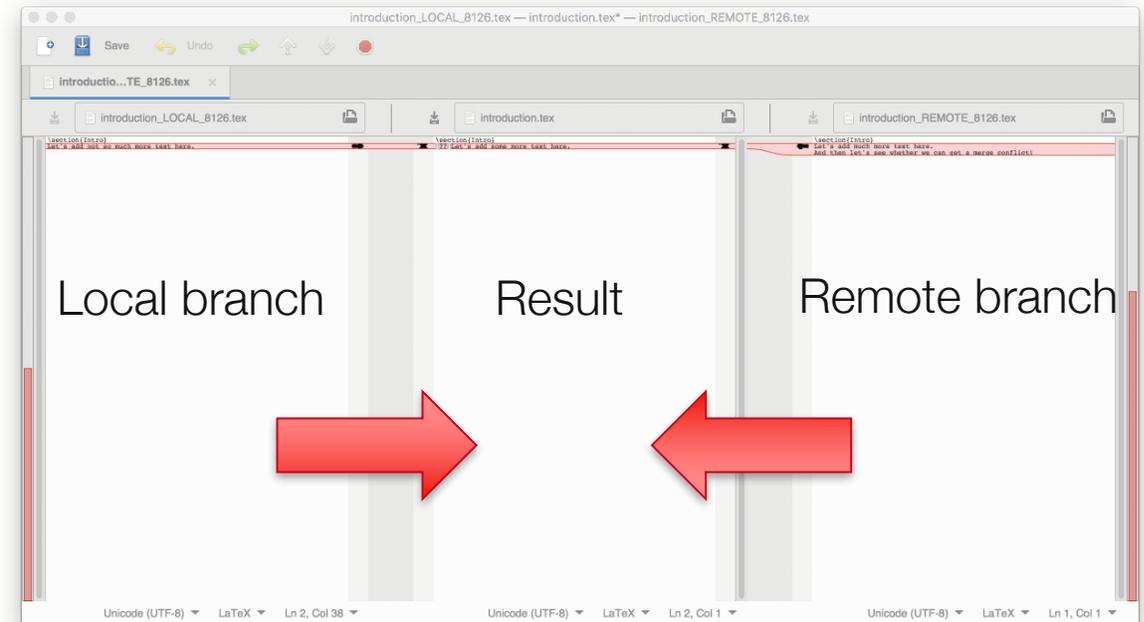
Merging to different branches

- After a few commits in your new branch (and master) we want to update the new branch with the changes in master
- In general, there are two options: merge, or rebase
 - For whatever we will encounter, rebase will do!



Merge conflicts

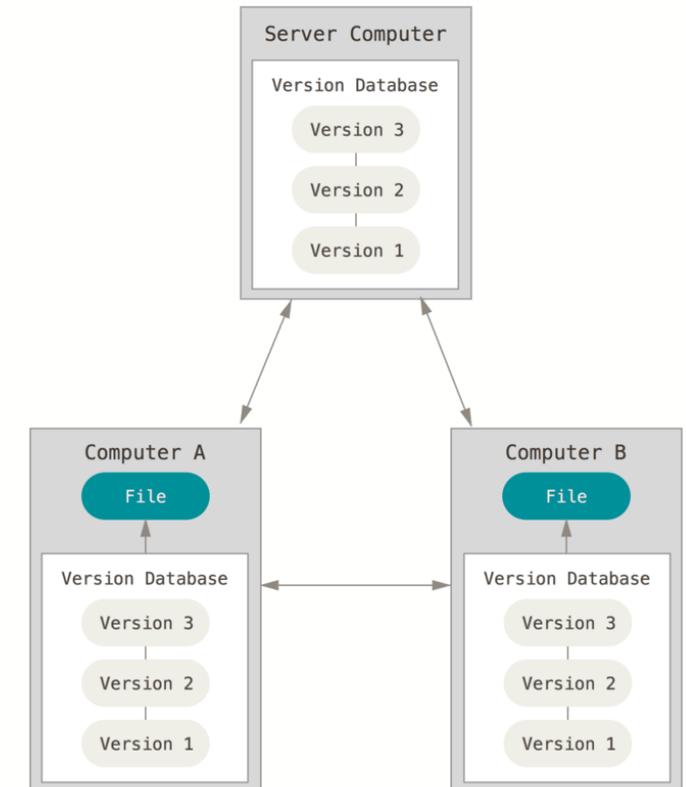
- Imagining you have modified one file in the two branches and want to rebase on master
- `git rebase master`
- git tries to figure out which part has been modified and whether it can put the stuff together
 - Sometimes this is not possible - merge conflict!
- `git mergetool`
 - Will open the configured merge tool
 - I recommend meld
 - Mac: `brew cask install meld`
 - Ubuntu: `sudo apt-get install meld`
 - `git config --global merge.tool meld`
- Once you're sure everything's fixed
 - `git rebase --continue`



Collaborative working – the ALICE example

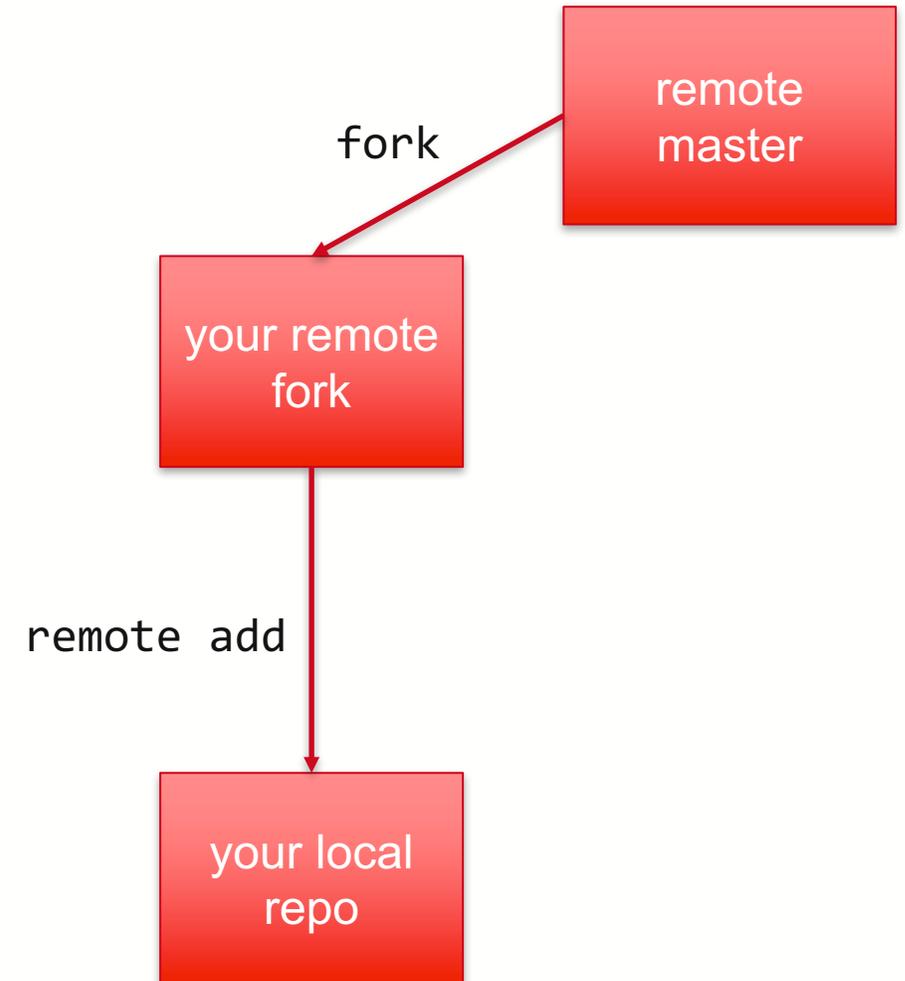
see also <https://alisw.github.io/git-tutorial/>

- The basic idea here is that we create a copy (fork) of the global project
 - We can play around freely without affecting the original project!
- I set up a dummy project on github (where e.g. all ALICE software is hosted)
 - <https://github.com/tutorialBot/dummyProject>
- To start working on the project you just get a clone of that
 - `git clone https://github.com/tutorialBot/dummyProject.git`
 - Happy gitting, but how to get local modifications to the original project?
 - If everybody could just add like that modifications -> 🤩
 - Pull requests!



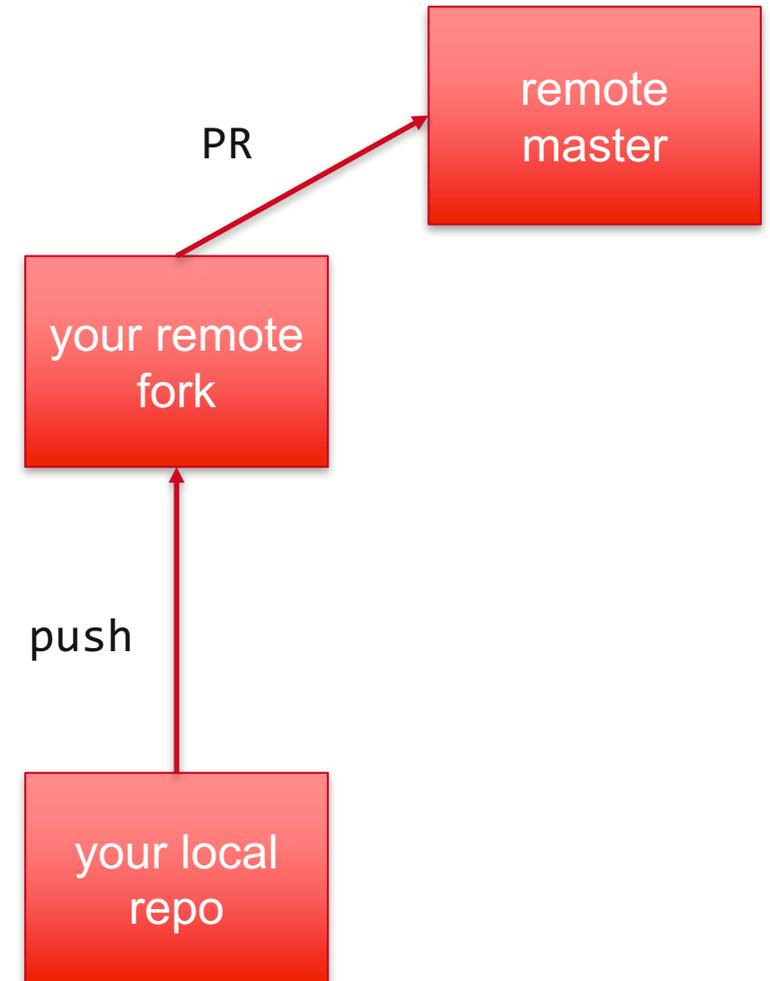
How not to mess up

- Get a copy (fork) of the project on github
 - Your local repo needs to know your fork exists on github
 - `git remote add <some name> <address of your fork>`



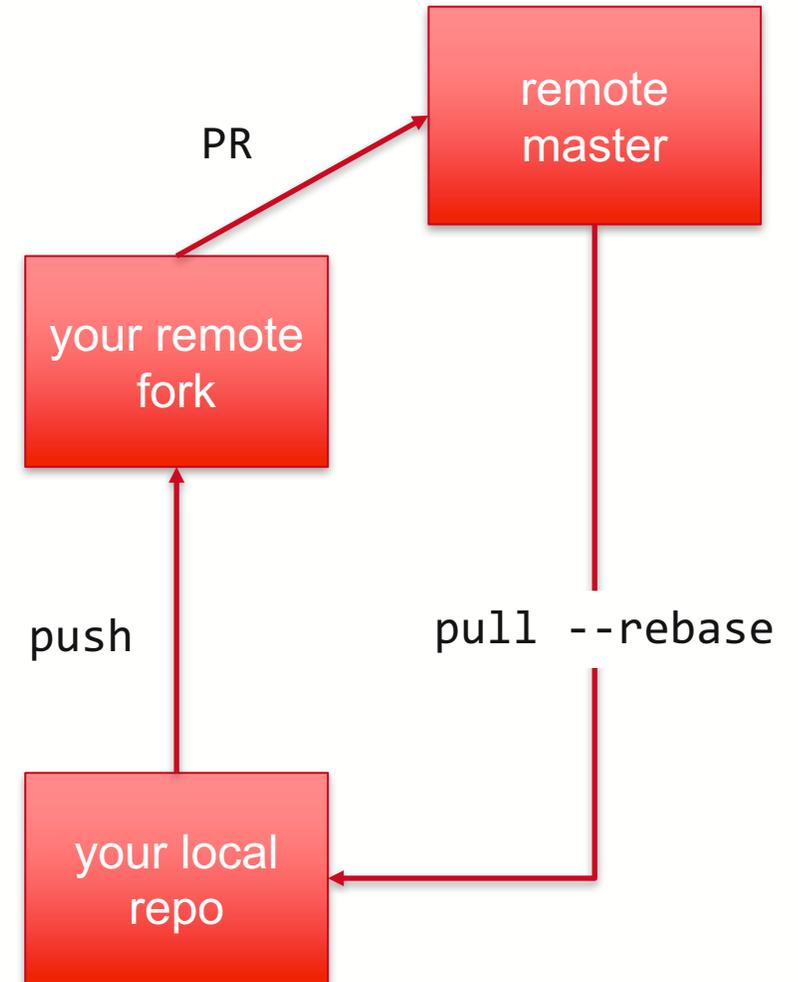
How not to mess up

- Get a copy (fork) of the project on github
 - Your local repo needs to know your fork exists on github
 - `git remote add <some name> <address of your fork>`
- Ideal workflow: work locally on a new branch!
 - `git checkout -b <new branch name>`
 - Do you changes and commit
 - Upload to your fork
 - `git push <remote name> <branch name>`
 - And then set up a pull request on github



How not to mess up

- Get a copy (fork) of the project on github
 - Your local repo needs to know your fork exists on github
 - `git remote add <some name> <address of your fork>`
- Ideal workflow: work locally on a new branch!
 - Do your changes and commit
 - Upload to your fork
 - `git push <remote name> <branch name>`
 - And then set up a pull request on github
- But... what if some changes on the master were made in between???
 - **IMPORTANT:** Update local repo before push!!!
 - `git pull --rebase <master repo> <branch name>`
 - `pull`: update local copy and merge
 - `-- rebase`: do a rebase instead of a merge



How to pull request

The screenshot shows the GitHub interface for a repository named 'a-mathis / dummyProject', which is a fork of 'tutorialBot/dummyProject'. The repository has 0 stars, 0 watches, and 1 fork. The 'Pull requests' tab is selected, showing 0 pull requests. A yellow highlight is placed over the 'Compare & pull request' button for the 'newBranch' branch, with a red arrow pointing to it from the right. Below this, the 'Branch: master' dropdown is visible, along with buttons for 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The repository description states 'No description, website, or topics provided.' and lists 3 commits, 3 branches, 0 releases, and 1 contributor. A list of files is shown, including 'introduction.tex' and 'paper.tex', both committed 20 hours ago. At the bottom, there is a prompt to 'Add a README'.



How to pull request

tutorialBot / dummyProject Watch 0 Star 0 Fork 1

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#)

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base fork: tutorialBot/dummyProject base: master ← head fork: a-mathis/dummyProject compare: newBranch

✓ **Able to merge.** These branches can be automatically merged.

 Title of your PR

Write Preview AA B i “ <> ↻ ☰ ☷ ☰ ↻ @ 📌 ↶

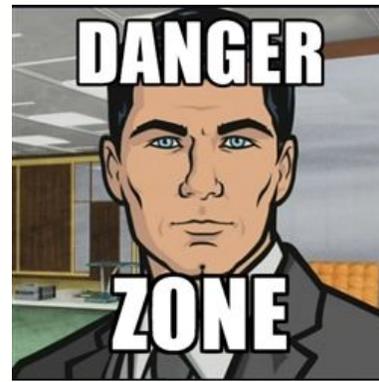
Why not write some meaningful comment here?

Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

Allow edits from maintainers. [Learn more](#) Create pull request



Advanced: How to mess around in the history



- If the project is well maintained (AliPhysics is not), PR's are reviewed
 - = most likely you will need to change some of the commits - how to do that?
 - Interactive rebase!
 - `git rebase -i @~X`, where `X` is the number of commits you want to go back to
 - Awesome, all information is there, when you run the command!
- To fix a commit: **pick**, change, **add** and **amend** the commit
 - Then, `git rebase --continue`
 - Can you just push to your remote? No!
 - Now you have to overwrite your remote – a forced push!
 - `git push -f <remote name> <branch name>`
 - As all forced operations: Potential screw-up...

Happy gitting!



**IN CASE
OF
FIRE**

- 🔗 `git add`
`git commit`
- ⬆️ `git push`
`git help push`
`git pull`
`git merge`
`git help reset`
`git reset --hard`
- 🔥 `nevermind, I might
as well burn
now...`