Git and Github seminar

Daniel de Andrés Hernández, dpto: Física Experimental de Altas Energías
Tomás Sánchez Sánchez-Pastor, dpto Computación Científica

GOBIERNO DE ESPAÑA

MINISTERIO DE ECONOMÍA, INDUSTRIA Y COMPETITIVIDAD

Ciemat

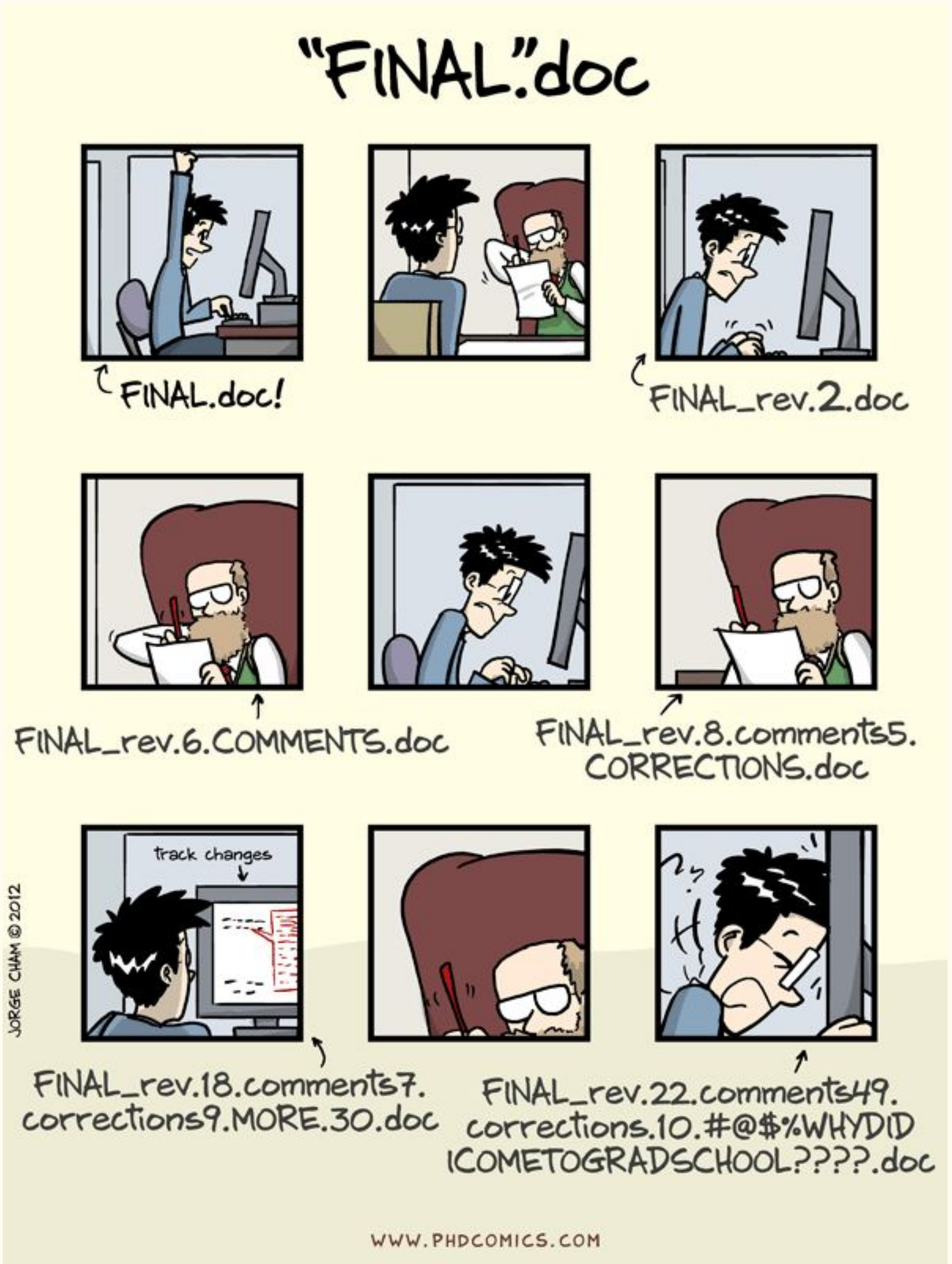
Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

Git and GitHub

in a nutshell

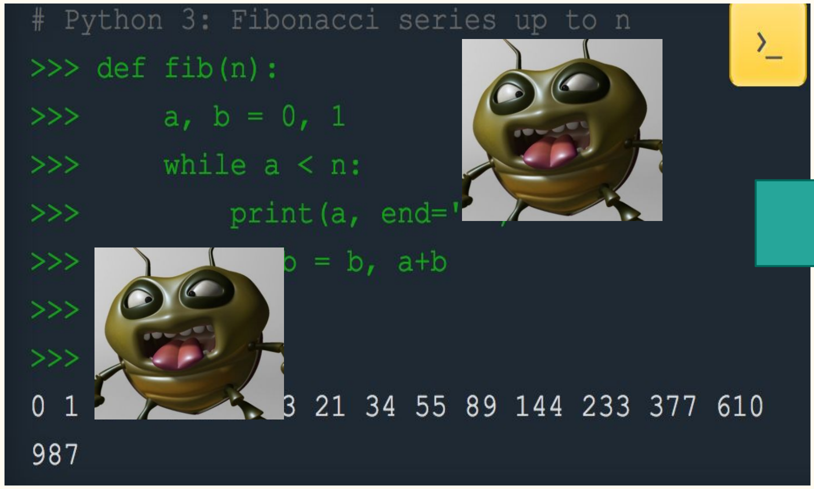
- 1. Basics
  - Git locations
  - Commit model
  - Branching
  - Merging
- 2. Advanced functions:
  - Rebase
  - Stash
- 3. Workflows

BASICS

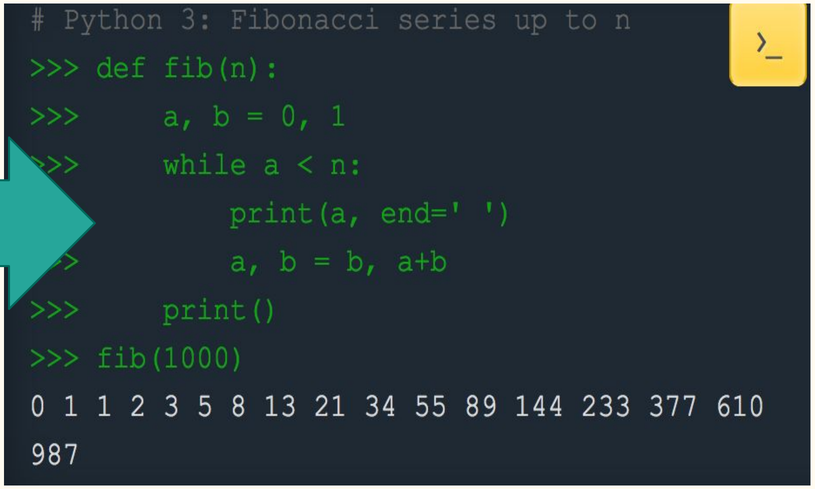

"FINAL".doc
FINAL.doc!
FINAL_rev.2.doc
FINAL_rev.6.COMMENTS.doc
FINAL_rev.8.comments5.
CORRECTIONS.doc
track changes
FINAL_rev.18.comments7.
corrections9.MORE.30.doc
FINAL_rev.22.comments49.
corrections.10.#@$%WHYDID
ICOMETOGRADSCHOOL????.doc
JORGE CHAM © 2012
WWW.PHDCOMICS.COM

Git and GitHub in a nutshell

- It allows us continuous improvement of our project
- Each **commit** is a snapshot of the project at a given time
- Git is very efficient
- We can review the project history (the commits) and undo the changes.



```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end='
>>>             b = b, a+b
>>>
>>>
0 1              3 21 34 55 89 144 233 377 610
987
```



```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
987
```

20 Files

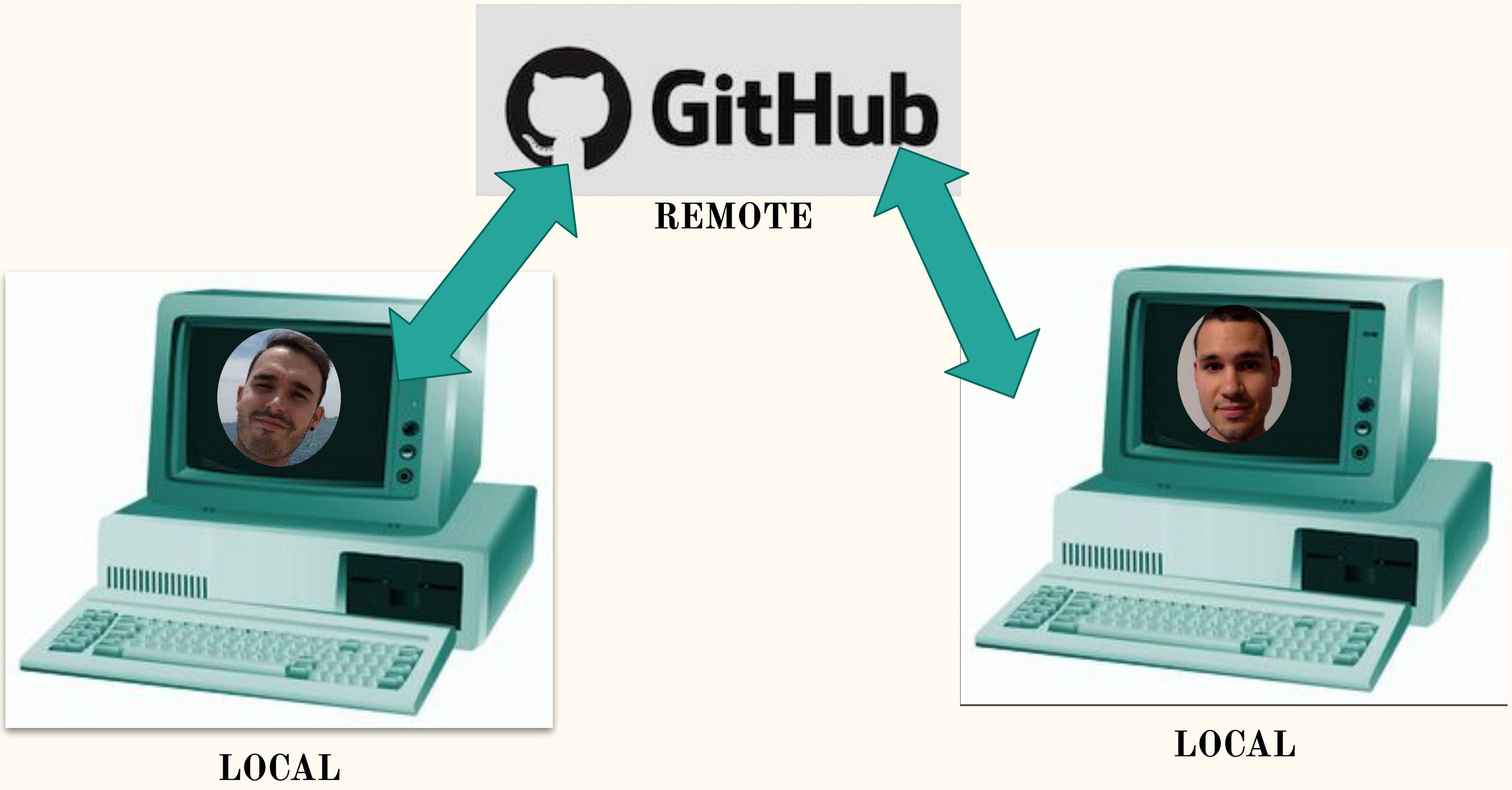
COMMITS

21 files

1 new file is store

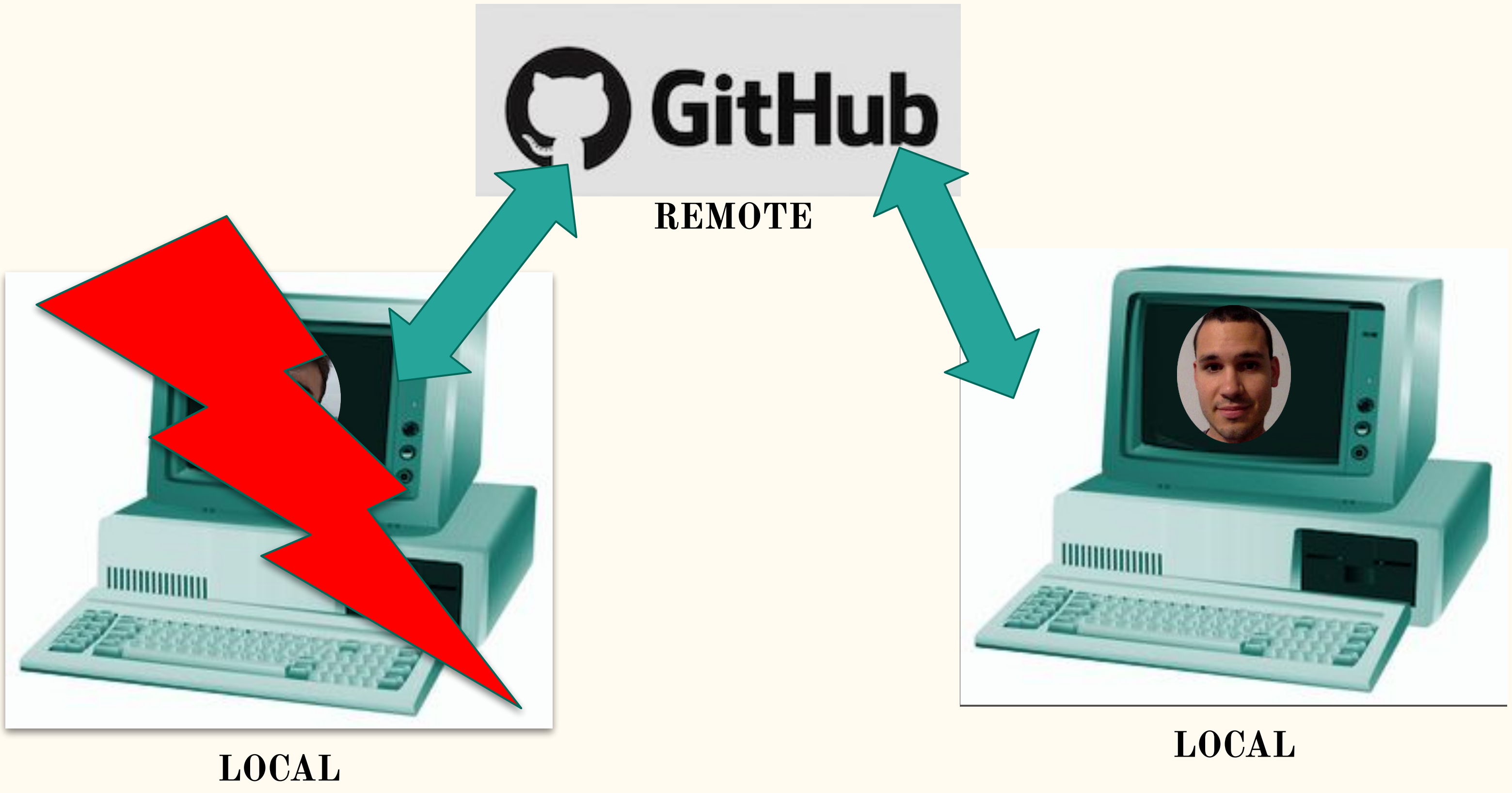
Git is a distributed version control system

- Each user has a local project history (repository)
- There is a single remote repository that is consider the source of truth
- Easily synchronise: pull and push commands



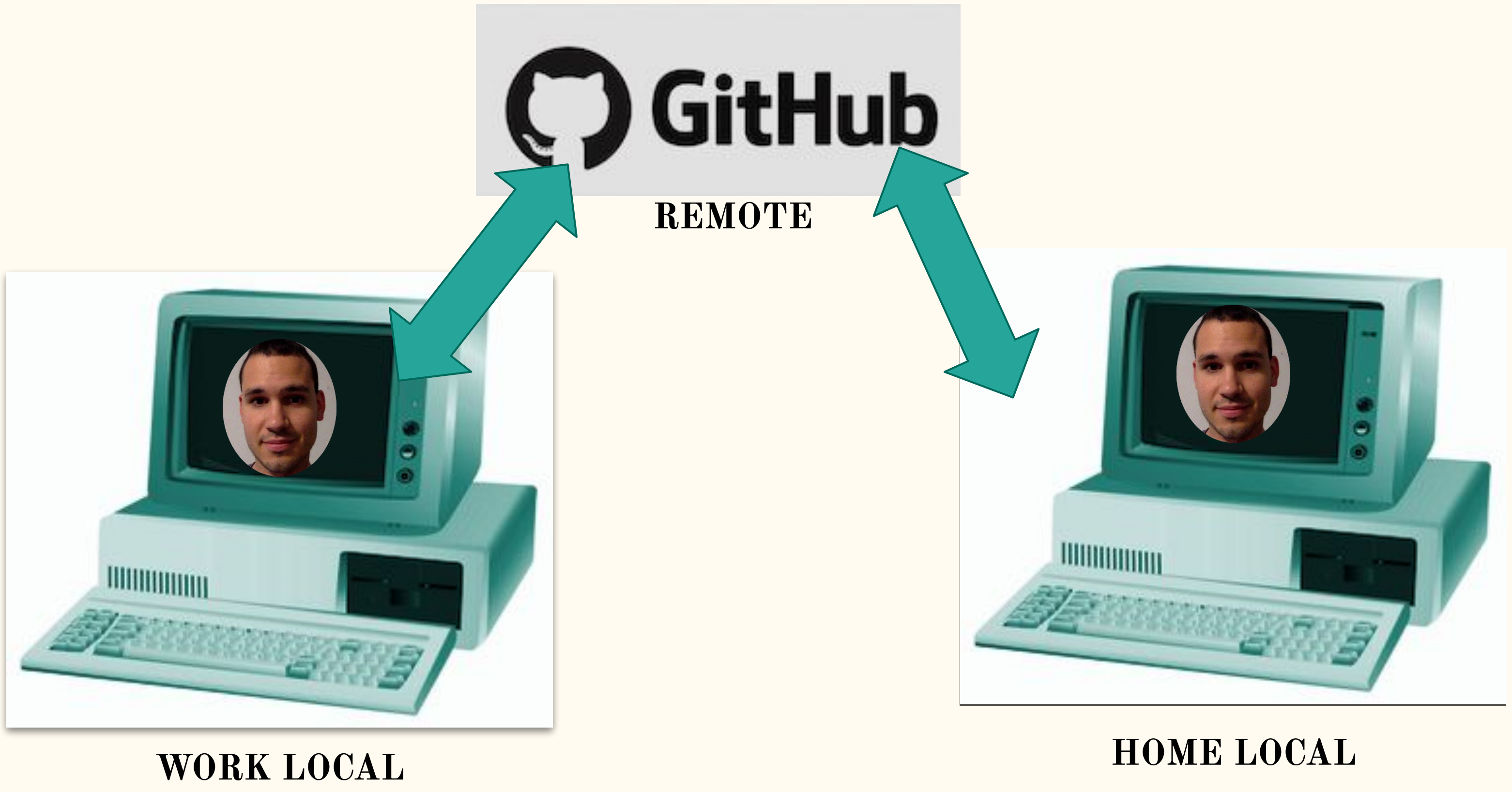
Git is a distributed version control system

- Each user has a local project history (repository)
- There is a single remote repository that is consider the source of truth
- Easily synchronise: pull and push commands



Git is a distributed version control system

- Each user has a local project history (repository)
- There is a single remote repository that is consider the source of truth
- Easily synchronise: pull and push commands

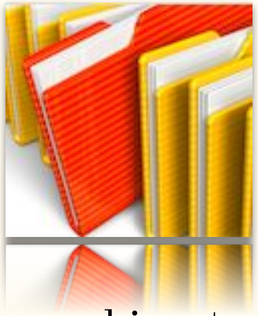


Git locations

- **Working Tree**
- **Staging area / index**
- **Local repository**
- **Remote repository**

Git locations

- **Working Tree :** A directory in your computer that contains the files of a single commit



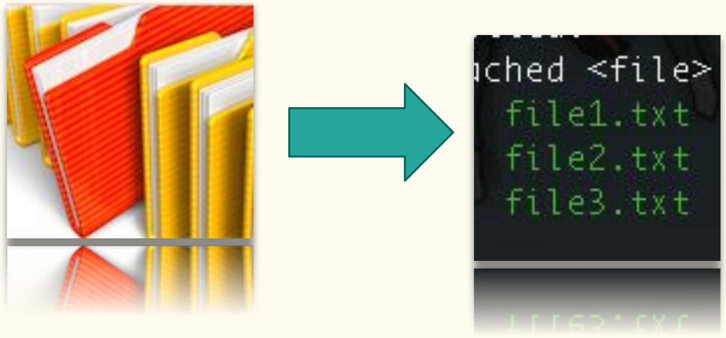
working tree

It is simple the directory where the files are located

You can access to the working tree (i.e. the files, code, etc) of every commit

Git locations

- **Working Tree :** A directory in your computer that contains the files of a single commit
- **Staging area / index :** Files that will be the next commit



```
ched <file>
  file1.txt
  file2.txt
  file3.txt
```

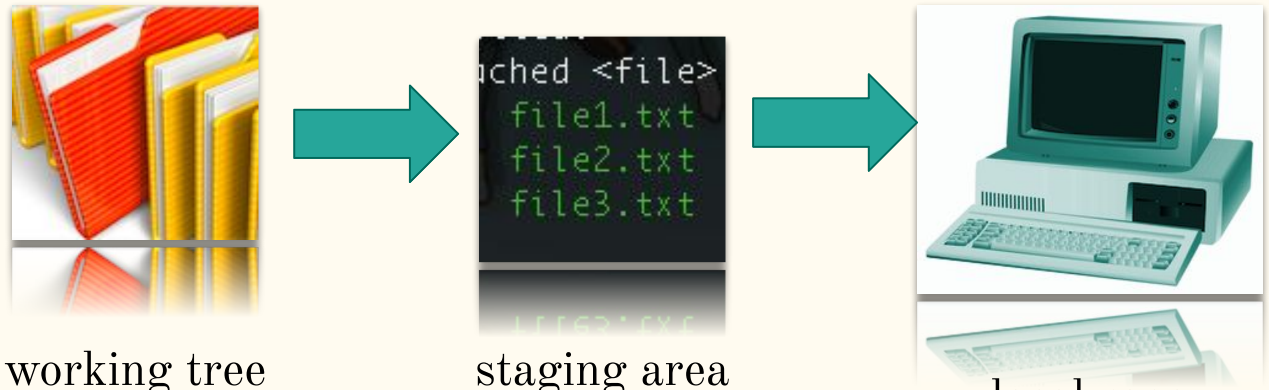
working tree

staging area

use the command **git add** [file] to select the files which are going to be added in the next commit

Git locations

- **Working Tree :** A directory in your computer that contains the files of a single commit
- **Staging area / index :** Files that will be the next commit
- **Local repository :** Contains all the commits of the project

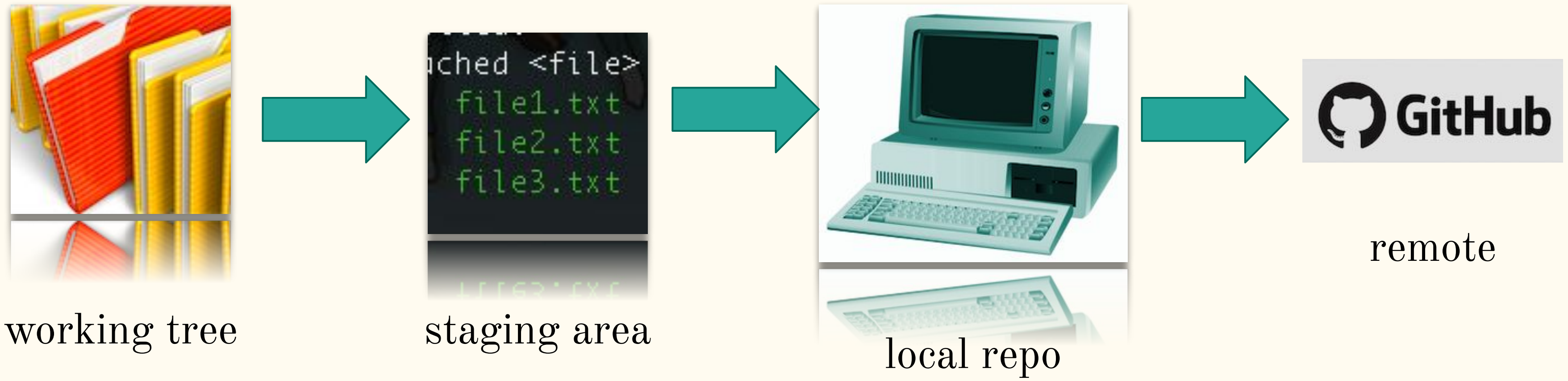


```
ched <file>
  file1.txt
  file2.txt
  file3.txt
```

add the commits to the local repository using **git commit -m" message"**

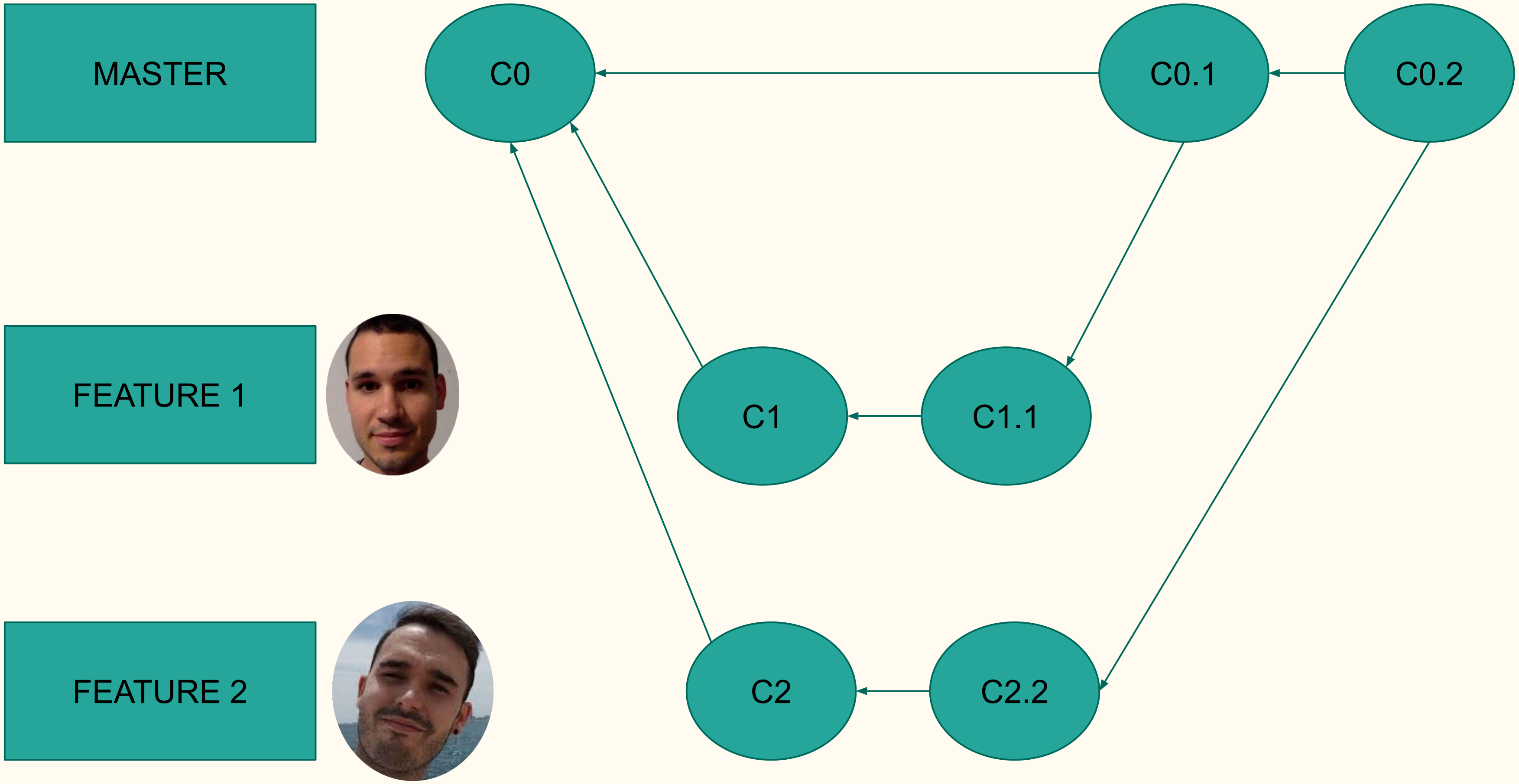
Git locations

- **Working Tree :** A directory in your computer that contains the files of a single commit
- **Staging area / index :** Files that will be the next commit
- **Local repository :** Contains all the commits of the project
- **Remote repository :** Contain the truth-commits of the project


```
ched <file>
  file1.txt
  file2.txt
  file3.txt
```

upload and download with **git push** and **git pull** respectively.

Git commit model



Hands on : git basics

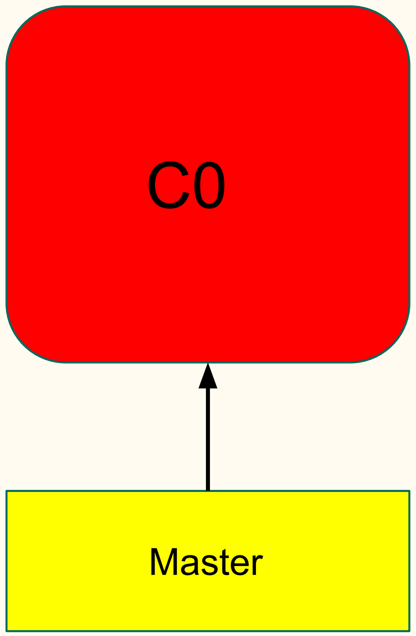
1. REPOSITORIES
- Create a remote repository using GitHub
- Clone a remote repository
- Create a local repository
- Upload your local content to a remote repository

2. COMMIT AND LOCATIONS

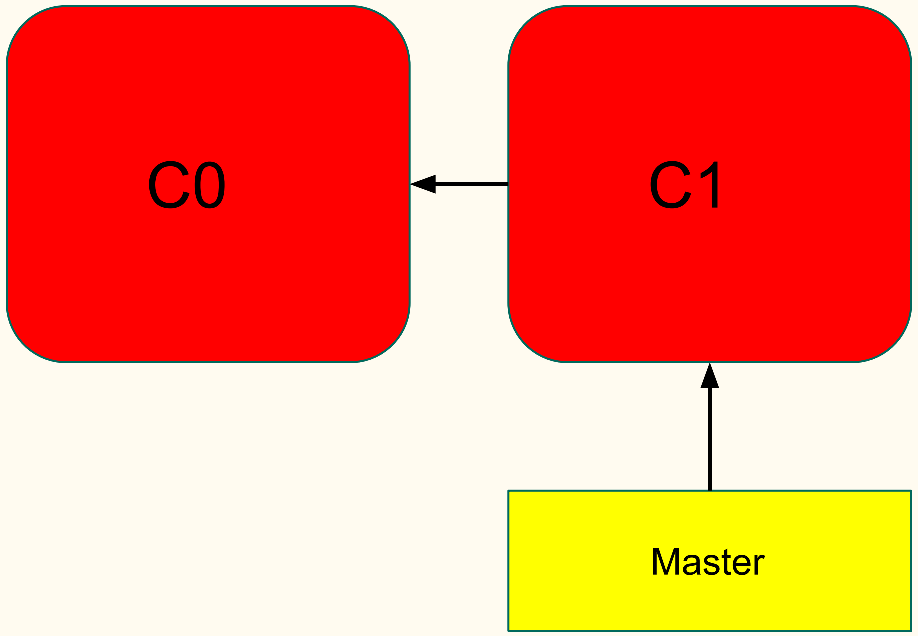
- Familiarize with commit model
- SHA-1s in Git
- Commit to a local repository
- Push to a remote repository
- Retrieve an older commit

ADVANCED

Branching: branches as movable pointers to commits



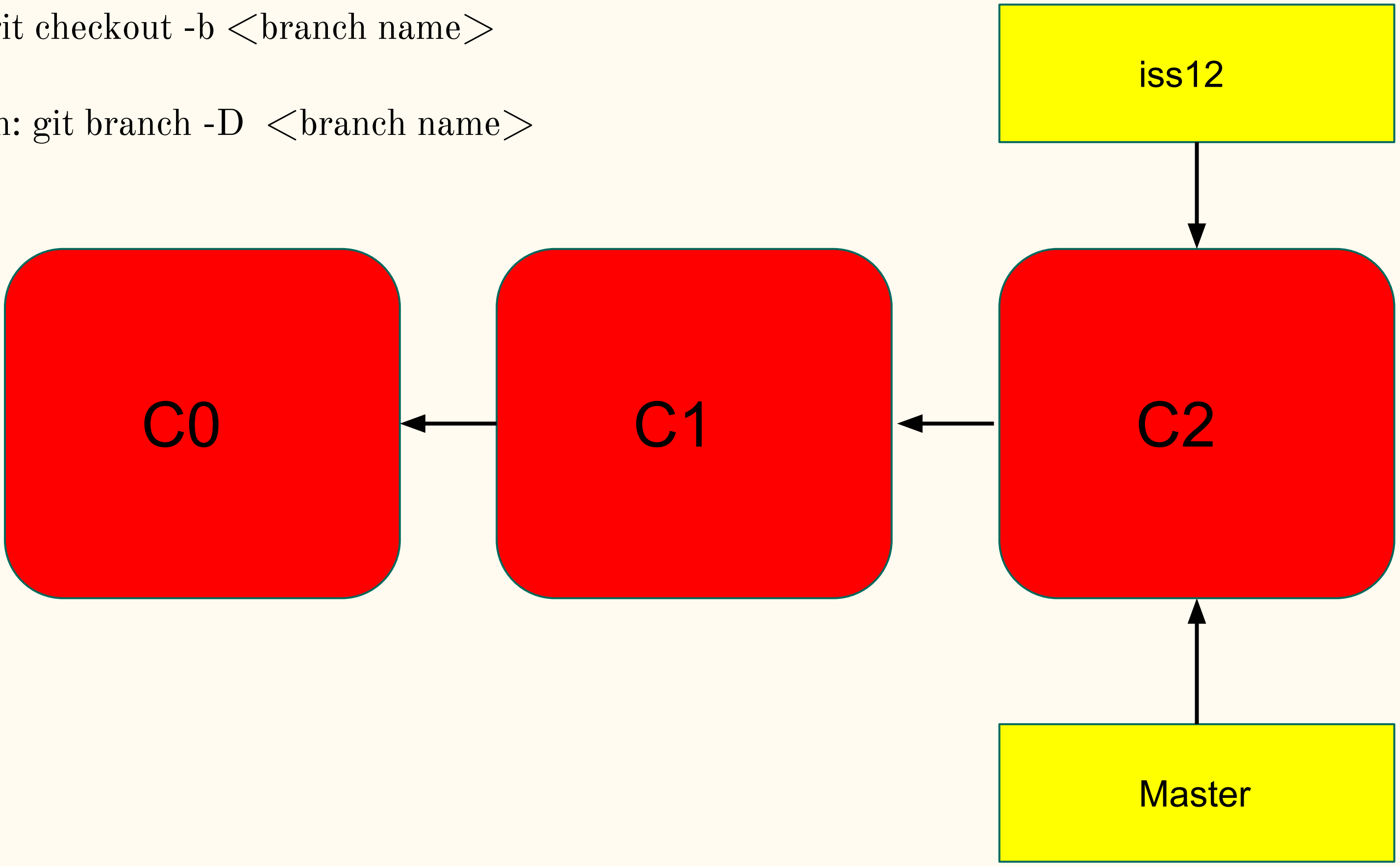
Branching: branches as movable pointers to commits



Branching: branches as movable pointers to commits

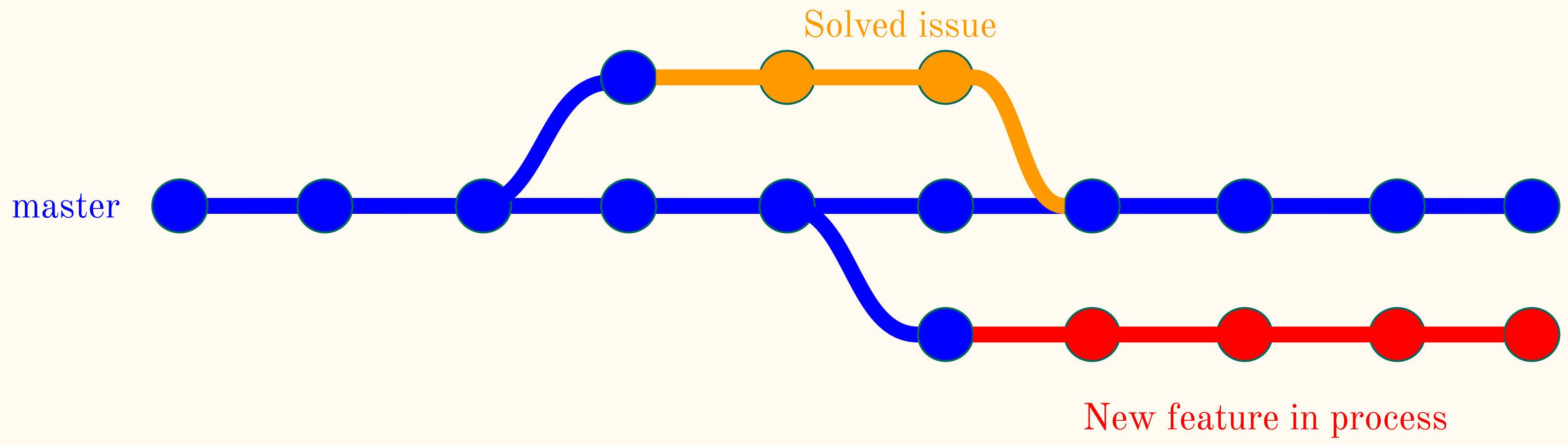
Creation: git checkout -b <branch name>

Destruction: git branch -D <branch name>



Branching

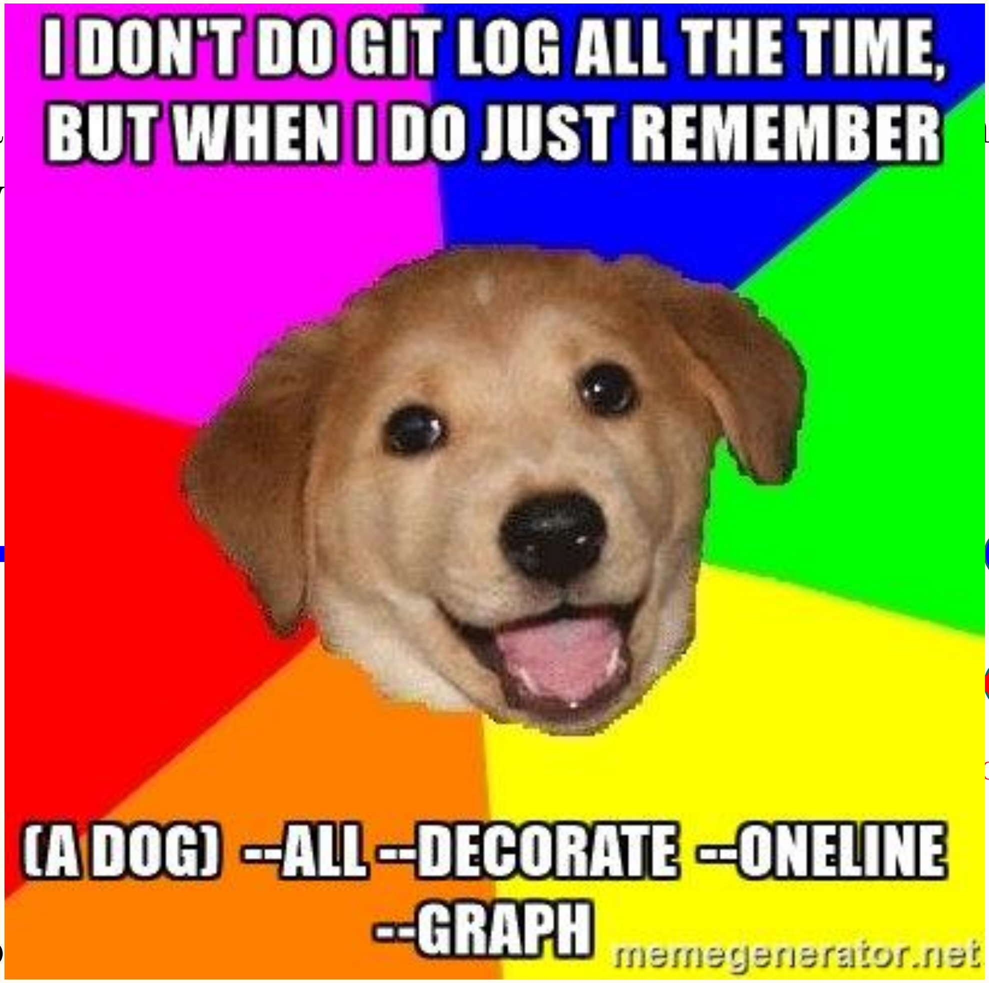
- The master branch is not special, is just the default name when you **git init** a repository. Nevertheless, for convenience, master must be your reference.



- Try **git adog** to show a graph of the branches history.

Branching

- The master bra you **git init** a repository. Nev reference.

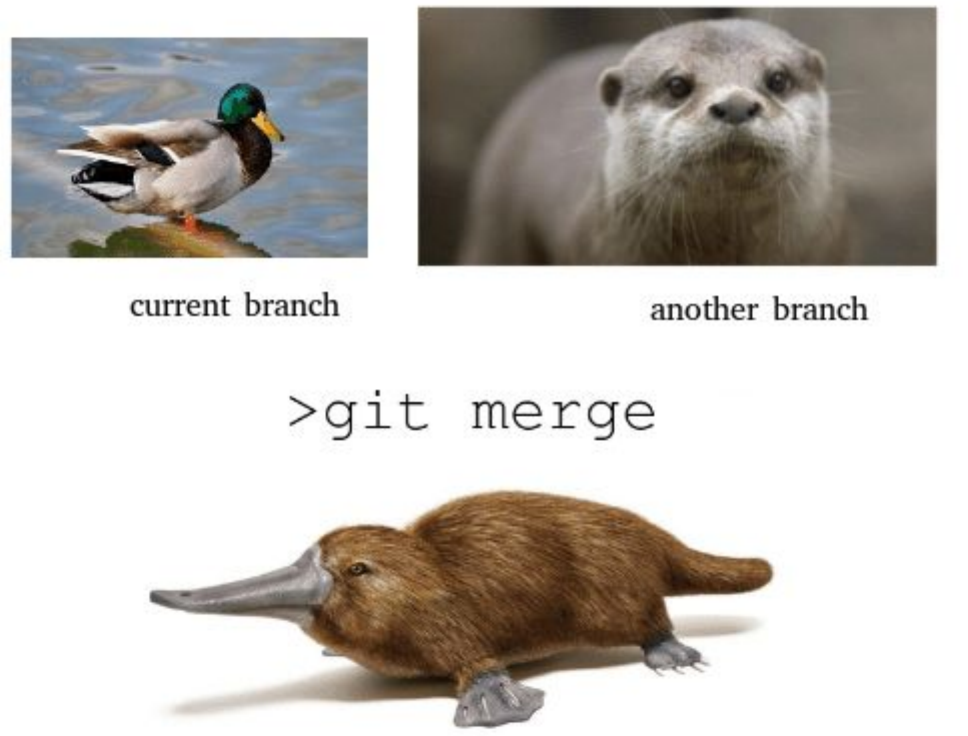

I DON'T DO GIT LOG ALL THE TIME, BUT WHEN I DO JUST REMEMBER
(A DOG) --ALL --DECORATE --ONELINE --GRAPH
memegenerator.net

- Try **git adog** to

Merging

- **git merge** is a fundamental operation applied to two branches that put together every change that has been made into a single branch.
- Warning! Merge is a source of conflicts!

Expectation



Reality

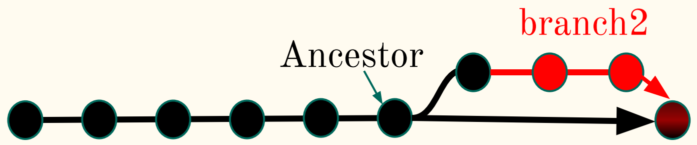
Merging

- **git merge** is a fundamental operation applied to two branches that put together every change that has been made into a single branch.
- Warning! Merge is a source of conflicts!

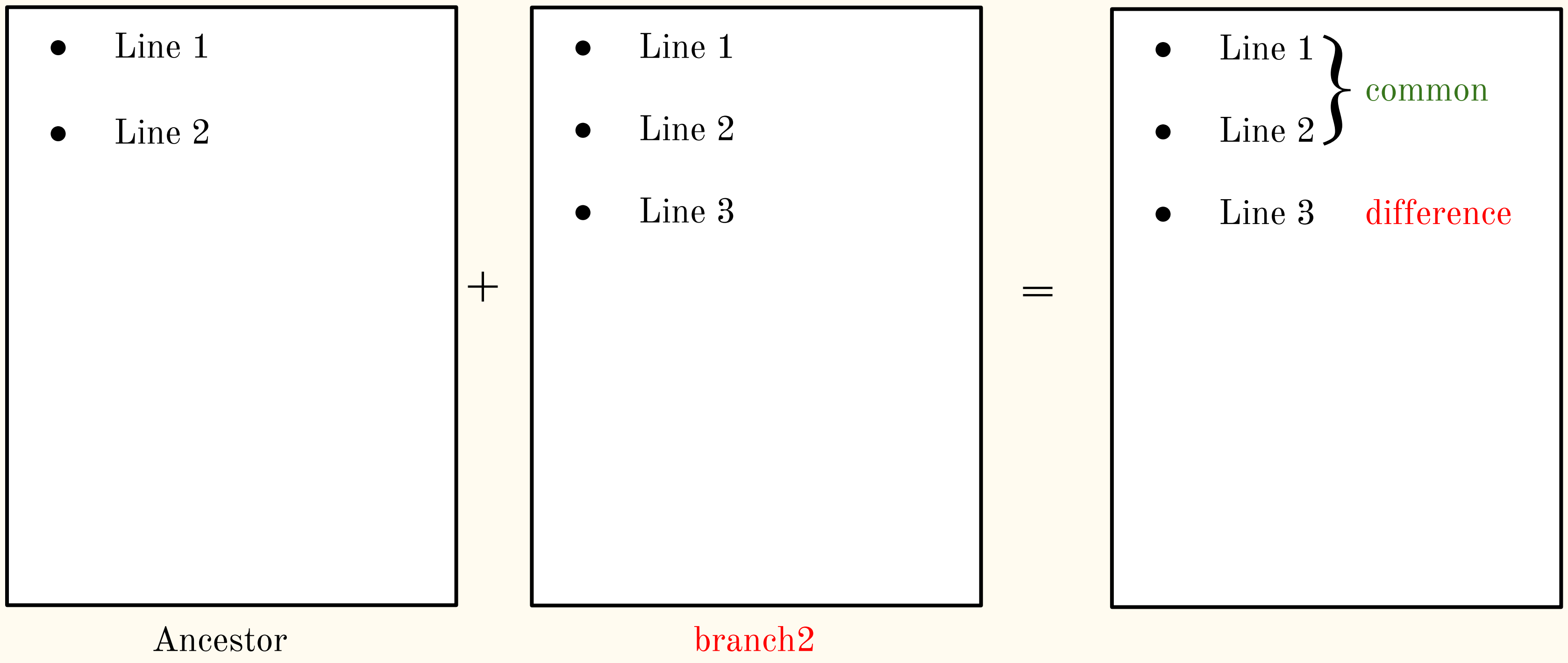
Expectation

Reality

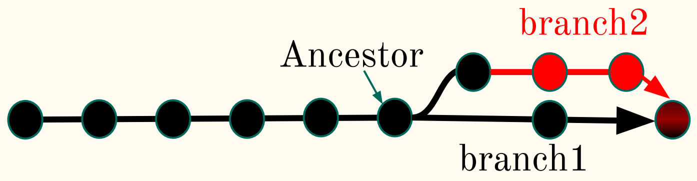
Merging: fast forward



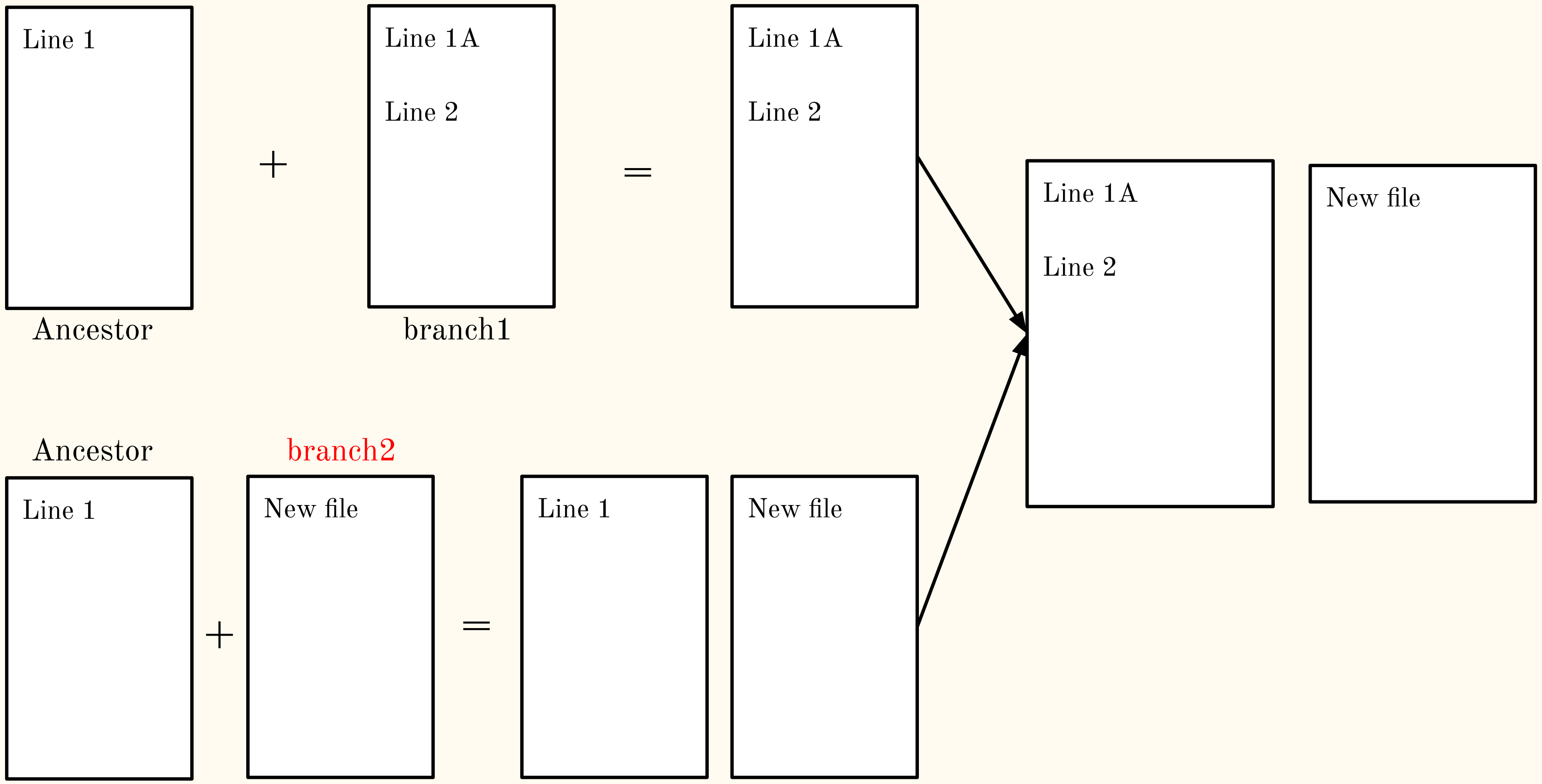
- When Git merges, firstly tries a fast forward.



Merging: 3-way merge



- If not possible, tries a 3-way merge.



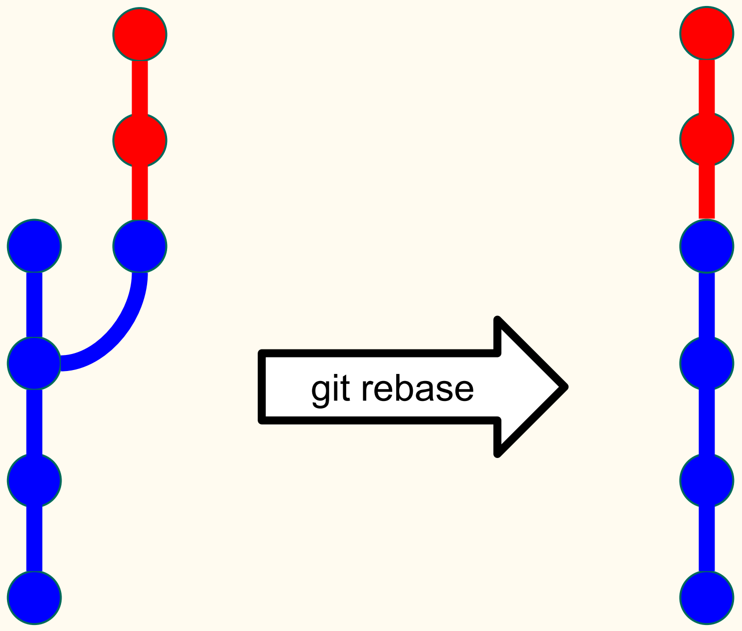
Merging: tips

- Merge conflicts happen when you merge branches that have competing commits.
- Before merge always check that you have pull the remote repository.
- Delete one of the merged branches.
- NEVER merge with master.

Advanced functions: Rebase

- git rebase <target branch> reapply commits on top of another branch.

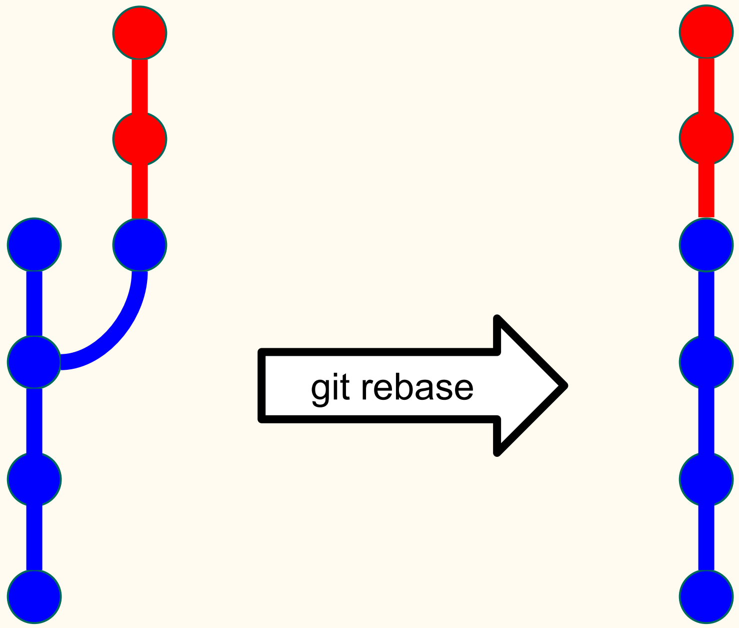
- Change the commit history and clean the graph.



Advanced functions: Rebase

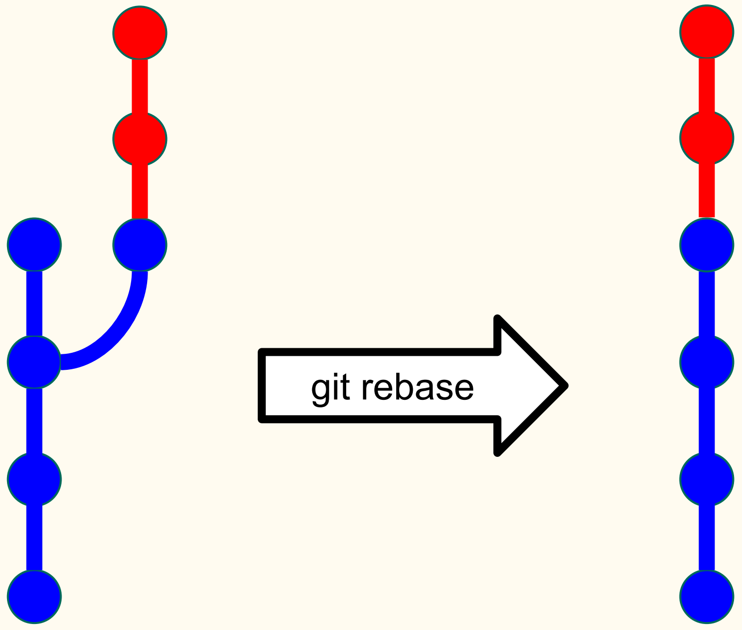
- git rebase <target branch> reapply commits on top of another branch.

- Change the commit history and clean the graph.



Advanced functions: Rebase

- git rebase <target branch> reapply commits on top of another branch.
- Change the commit history and clean the graph.

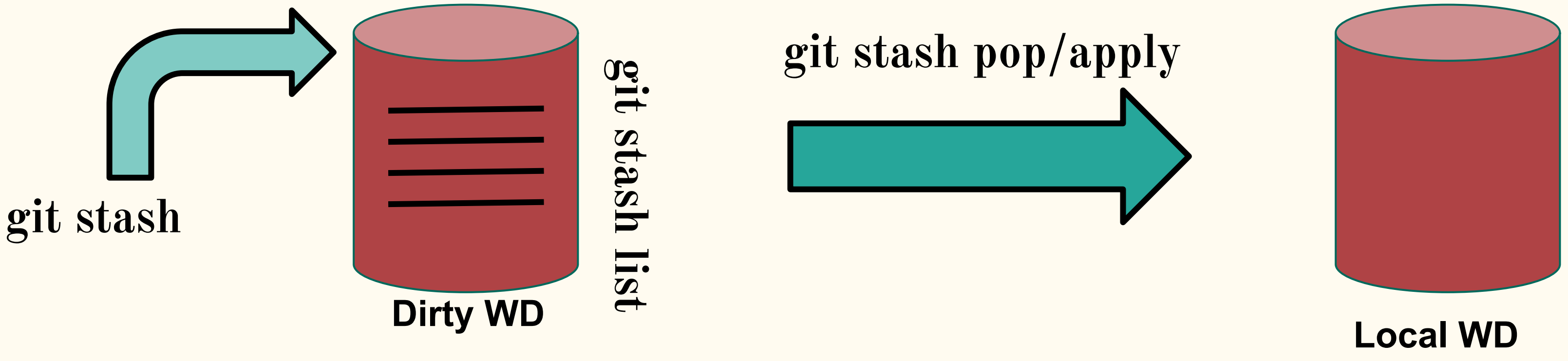


Advanced functions: Stash

- git stash saves the uncommitted changes in a dirty working directory and let you to reapply it whenever you want.

- Useful for solving quick bugs and for prevent pull errors.

- Main commands:



Advanced functions: Stash

- git stash saves the uncommitted changes in a dirty working directory and let you to reapply it whenever you want.

- Useful for solving quick bugs and for prevent pull errors.

- Main commands:

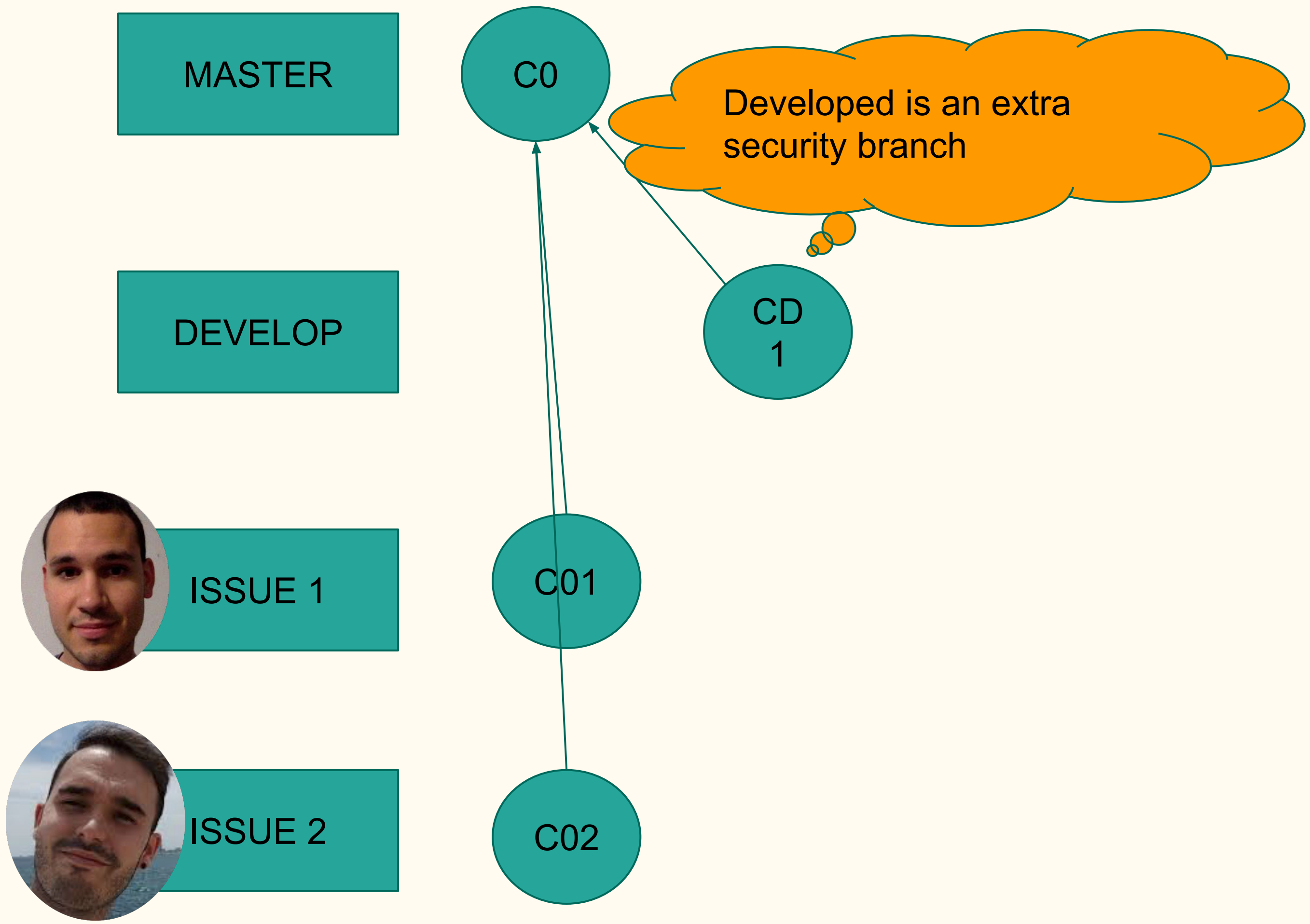
git stash clear

Dirty WD
Local WD

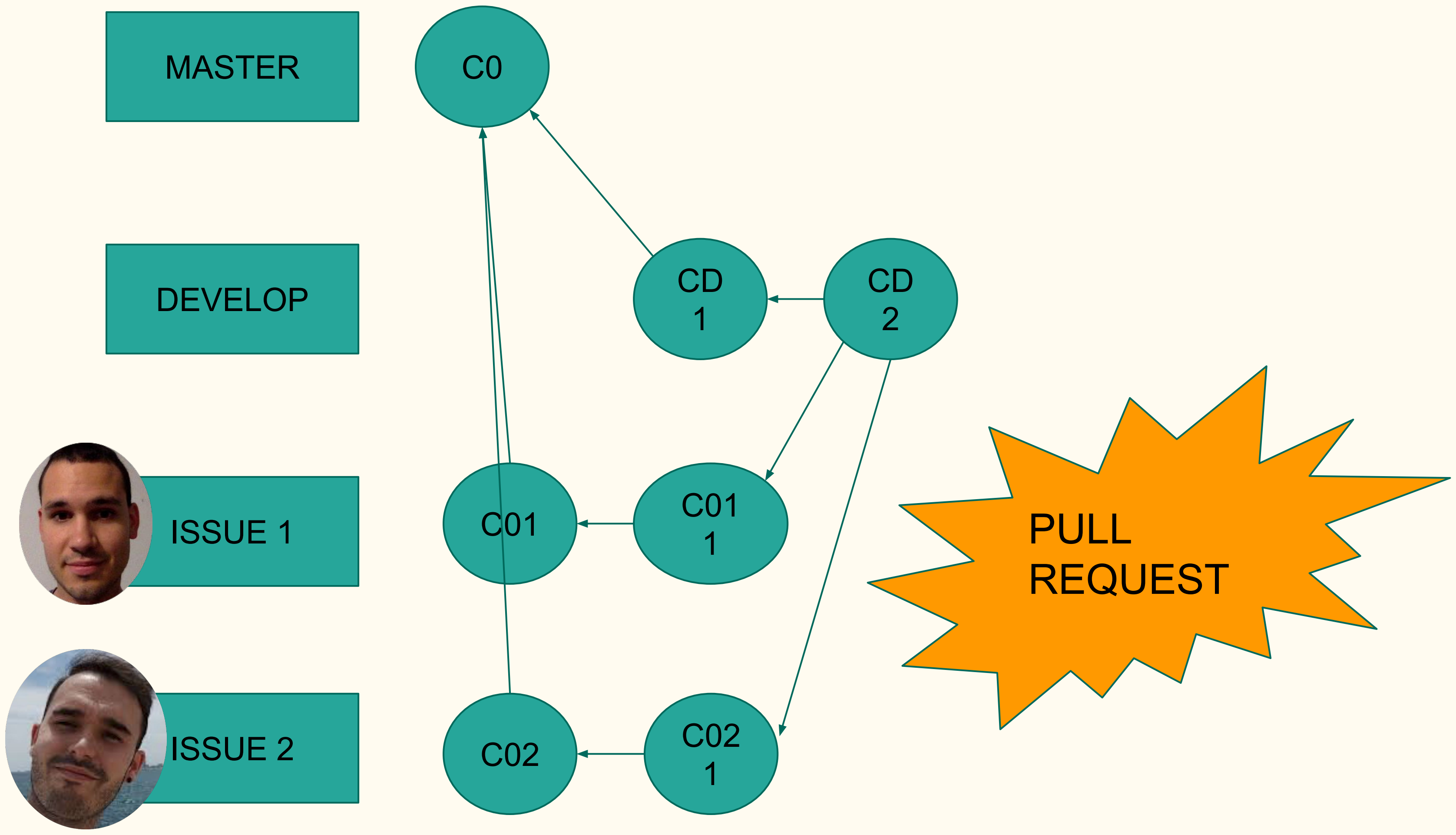
WORKFLOWS

Pull Requests and Issues

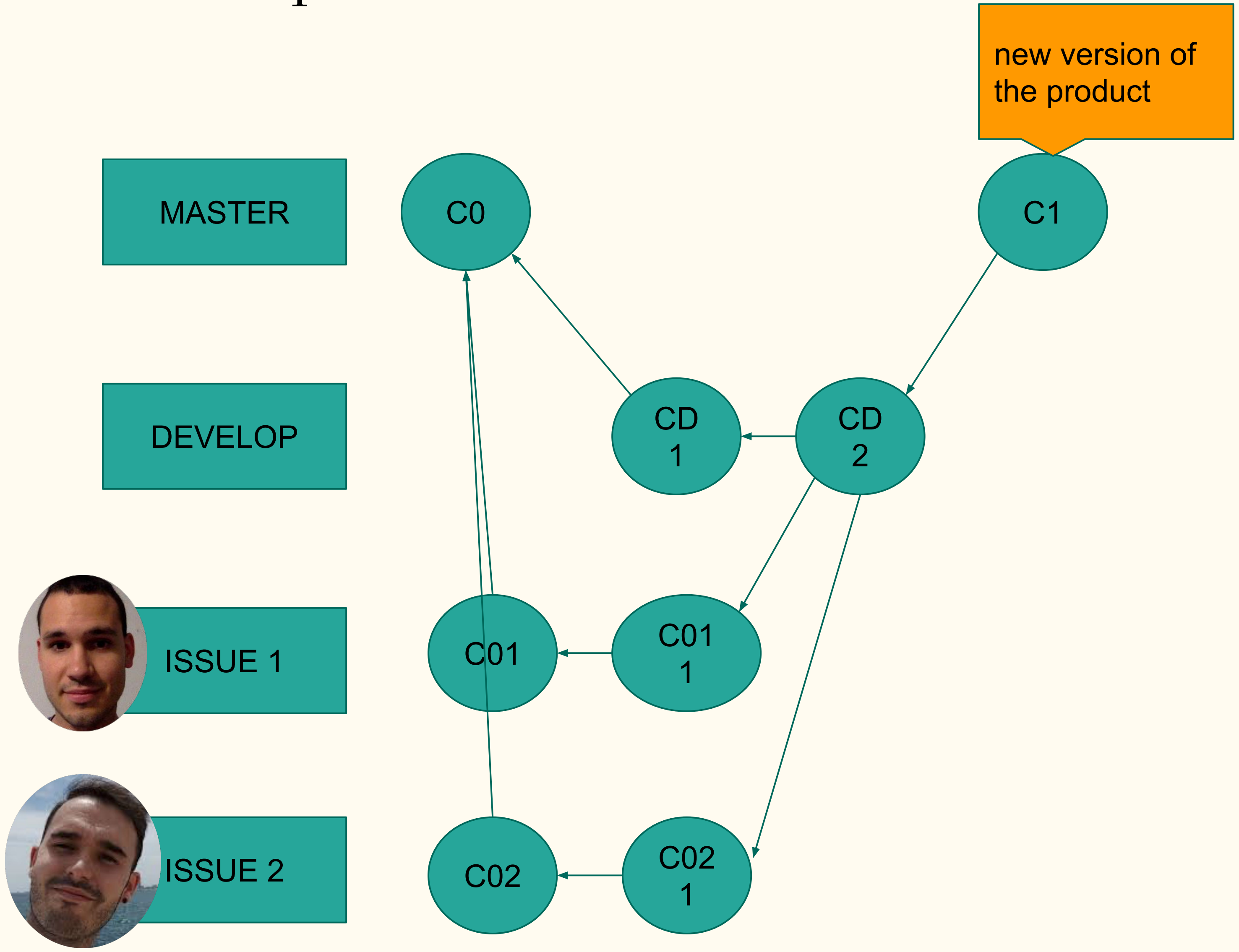
Pull Requests and Issues



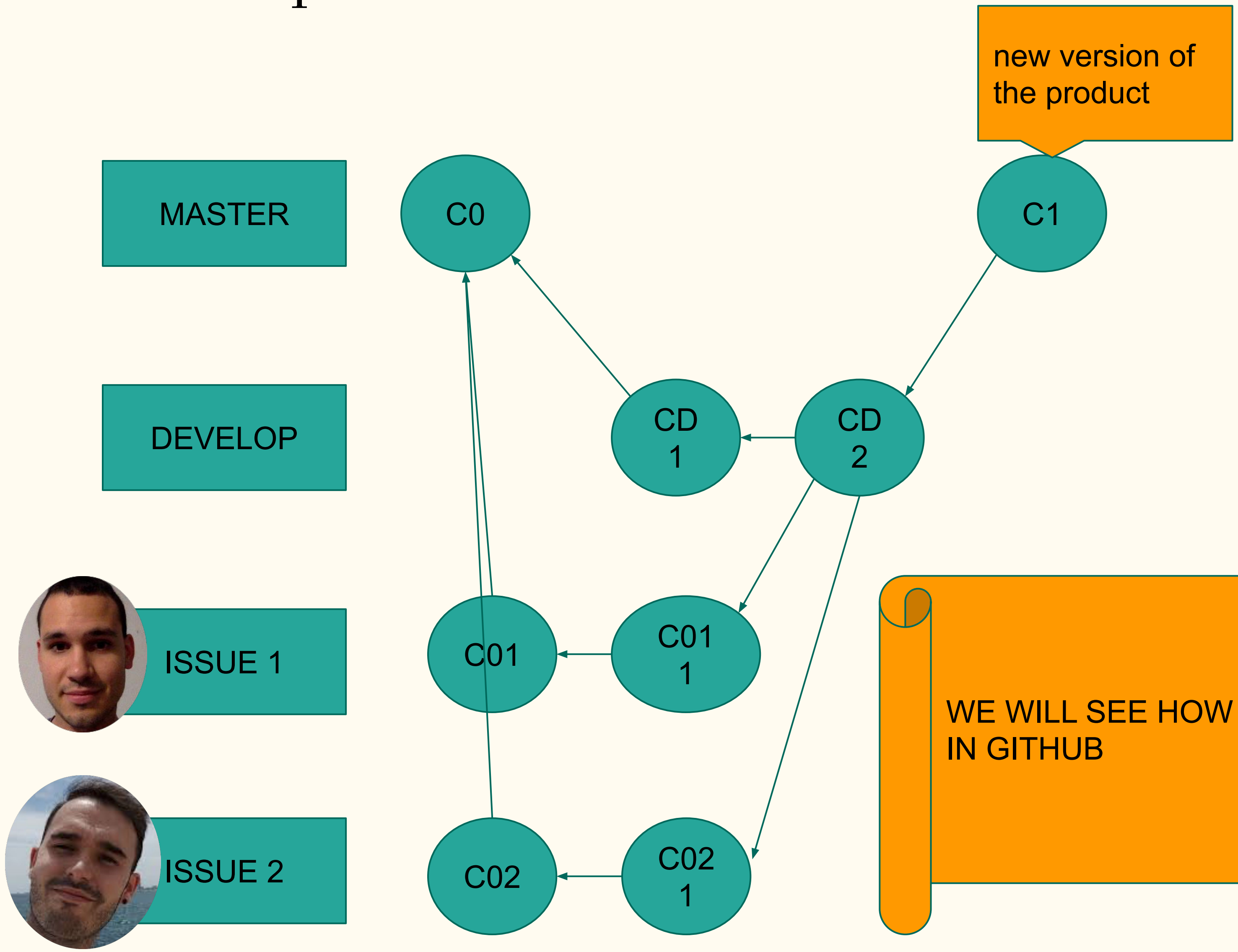
Pull Requests and Issues



Pull Requests and Issues



Pull Requests and Issues



WE WILL SEE HOW TO DO THIS IN GITHUB