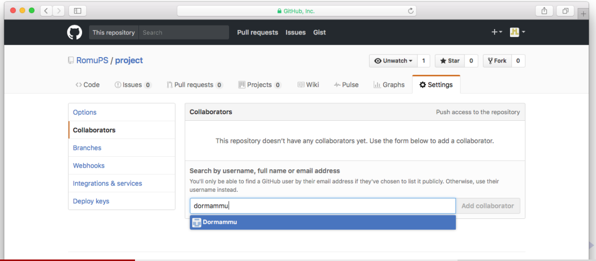
Collaborating with GitHub

GitHub allows the collaboration of different users on the same repository

There are two possibilities for collaboration:

- In the tab Settings, define a collaborator, by searching his/her username or email address. This gives the collaborator(s) full access to push data to the remote repository
- While this is simple, it can lead to problems
  - Two people trying to push the same file, with different versions
  - Can lead to conflicts, which are difficult to observe and fix, since everybody has the same privileges



Collaborators

Push access to the repository

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

dormammu

Cloning and forking

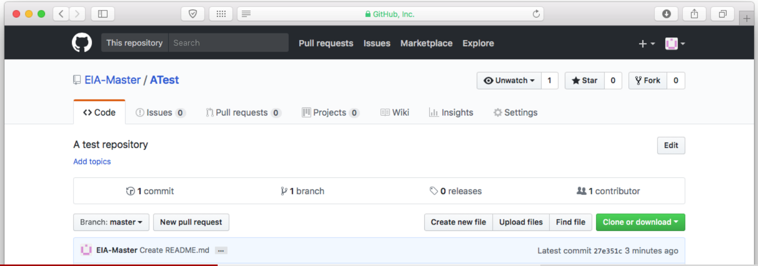
Another approach to collaborating on GitHub consists in **forking** an already existing project

To start the collaboration, some manager creates a main repository, where the main version of the code base will be maintained

Collaborators, on their GitHub page, locate the page of the manager, and the main repository.

On this page, collaborators click on the button `Fork`

This creates in your own GitHub account a full copy of the main repository



Cloning a remote repository

To have access to the forked repository we just created, we have to clone it, from our GitHub space, into our local machine (user your own github username!!)

```
:> git clone https://github.com/RomuPS/ATest.git
Cloning into 'ATest'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
:> cd ATest/
:ATest> ls -l
total 4
-rw-r--r-- 1 romu staff 30 Feb 23 16:06 README.md
:ATest> git remote -v
origin        https://github.com/RomuPS/ATest.git (fetch)
origin        https://github.com/RomuPS/ATest.git (push)
```

The cloning operation directly links our local to our remote copy of the main repository, with the alias "origin"

In this way, we can start making changes and updating our own version of the main repository

Updating a cloned remote repository

But in the mean time, the main repository can have been changed.

To update the changes in the main repository into our local version, we have to link it to the main, for which we choose the alias ”main”

```
:ATest> git remote add main https://github.com/EIA-Master/ATest.git
:ATest> git remote -v
main        https://github.com/EIA-Master/ATest.git (fetch)
main        https://github.com/EIA-Master/ATest.git (push)
origin        https://github.com/RomuPS/ATest.git (fetch)
origin        https://github.com/RomuPS/ATest.git (push)
```

Now we can fetch changes in main

```
:ATest> git fetch main
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/EIA-Master/ATest
   27e351c..f2cd819  master     -> main/master
```

Working in a cloned remote repository

We now work on our cloned repository, adding a file

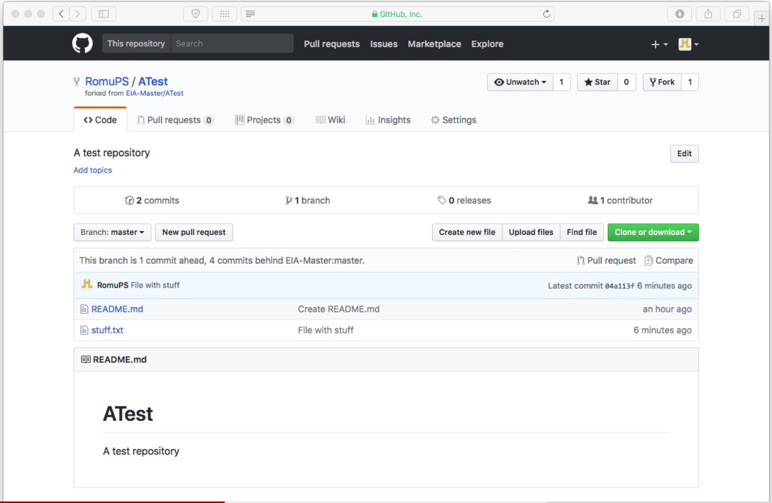
```
:ATest> vim stuff.txt
:ATest> cat README.md
A file with some stuff

:ATest> git add stuff.txt
:ATest> git commit -m "File with stuff"
[master 04a113f] File with stuff
 1 file changed, 2 insertions(+)
 create mode 100644 stuff.txt
# push to our cloned remote "origin"
:ATest> git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 313 bytes | 313.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/RomuPS/ATest.git
   27e351c..04a113f  master -> master
```

But since we don't know the password for the main repository "main", we cannot directly push there our changes

```
:ATest> git push main master
remote: Permission to EIA-Master/ATest.git denied to RomuPS.
fatal: unable to access 'https://github.com/EIA-Master/ATest.git/':
The requested URL returned error: 403
```

Pulling to the main remote repository

To pull changes to the main, we have to open a "New pull request" in our GitHub web page



A test repository

This branch is 1 commit ahead, 4 commits behind EIA-Master:master.

RomuPS File with stuff		Latest commit 04a113f 6 minutes ago
README.md	Create README.md	an hour ago
stuff.txt	File with stuff	6 minutes ago

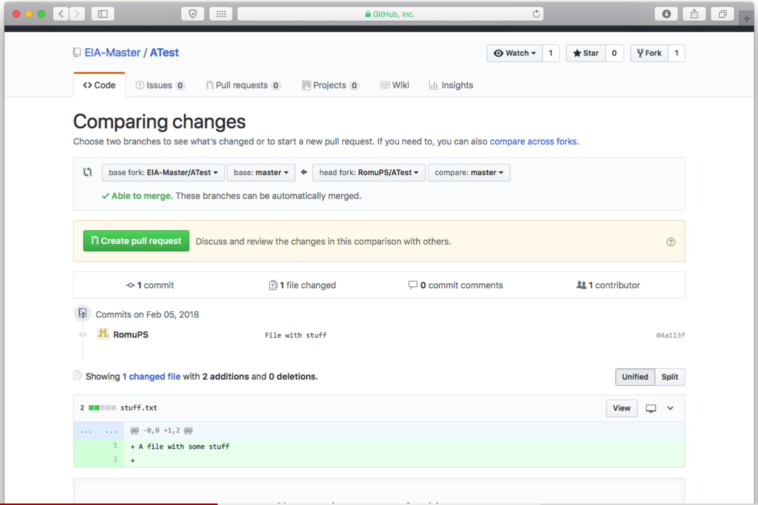
README.md

ATest

A test repository

Pulling to the main remote repository

We have to choose the base fork (the main) and our fork, choose the branches we want to pull, add a comment, and submit



Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

base fork: EIA-Master/ATest · base: master ← head fork: RomuPS/ATest · compare: master

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

Discuss and review the changes in this comparison with others.

1 commit · 1 file changed · 0 commit comments · 1 contributor

Commits on Feb 05, 2018

RomuPS   File with stuff   04a113f

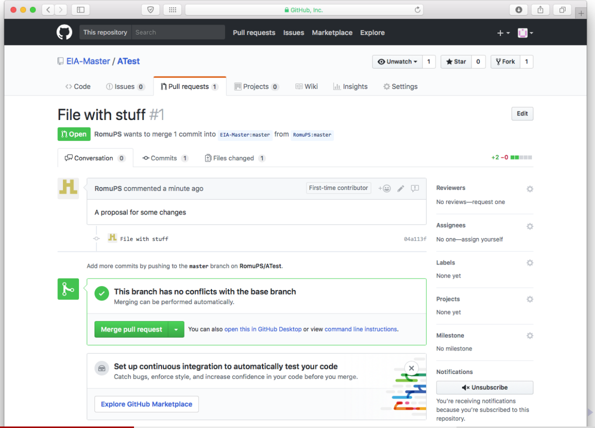
Showing 1 changed file with 2 additions and 0 deletions.

```
2 stuff.txt
... ... @@ -0,0 +1,2 @@
    1   + A file with some stuff
    2   +
```

Pulling to the main remote repository

In the GitHub page of the owner of main, appears a pull request, in the corresponding tab. The owner of main can check it, see if there are no conflicts, and accept (merge) it.

Both remotes (origin and main) will be now synchronized



File with stuff #1

RomuPS wants to merge 1 commit into `EIA-Master:master` from `RomuPS:master`

RomuPS commented a minute ago

A proposal for some changes

`File with stuff` `04a113f`

Add more commits by pushing to the **master** branch on **RomuPS/ATest**.

This branch has no conflicts with the base branch
Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.

Set up continuous integration to automatically test your code
Catch bugs, enforce style, and increase confidence in your code before you merge.

Reviewers
No reviews—request one

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

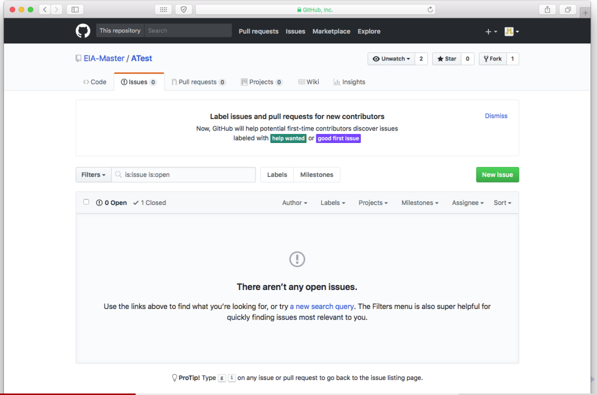
Milestone
No milestone

Notifications
You're receiving notifications because you're subscribed to this repository.

Managing collaboration

Apart from adding new pieces of code, GitHub allows discussion on problems (issues) in the codebase. To start a discussion, we have to open a new issue in the Issues tab of the main repository.

The discussion can be continued by the members of the collaboration, and closed when the problem raised has been solved.



Label issues and pull requests for new contributors

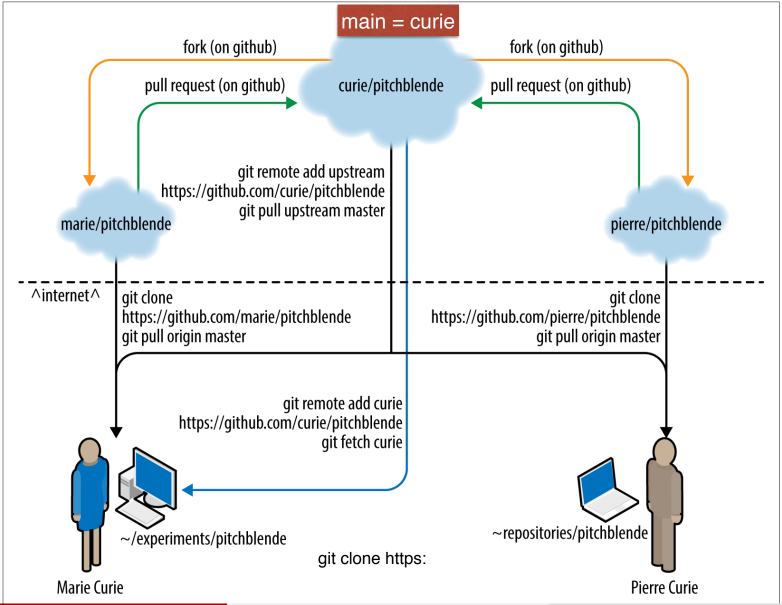
Now, GitHub will help potential first-time contributors discover issues labeled with help wanted or good first issue

There aren't any open issues.

Use the links above to find what you're looking for, or try a new search query. The Filters menu is also super helpful for quickly finding issues most relevant to you.

ProTip! Type g i on any issue or pull request to go back to the issue listing page.

Forking and cloning sum up



main = curie

```
git remote add upstream
https://github.com/curie/pitchblende
git pull upstream master
```

```
git clone
https://github.com/marie/pitchblende
git pull origin master
```

```
git clone
https://github.com/pierre/pitchblende
git pull origin master
```

```
git remote add curie
https://github.com/curie/pitchblende
git fetch curie
```

```
git clone https:
```

Collaboration exercise

We are going to develop a fake project in which I will play the role of boss.

- Go to my github page (RomuPS user) and fork and clone the repository `Project_mock_up`
- Link your own copy to my folder as main
- Open a branch of the project in your own local folder, add some source code for something simple and merge it into your master
- Make a pull request into my account
- Let us see what happens ...