Github basics

Ekta Patel
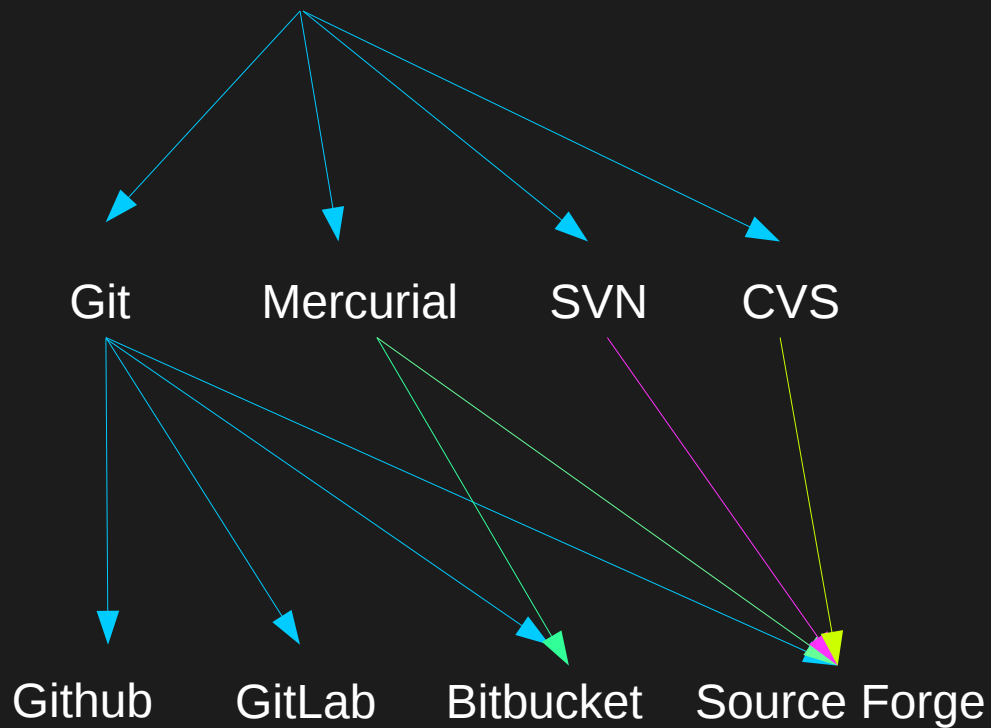
&

Nicolas Garavito-Camargo

Code Coffee
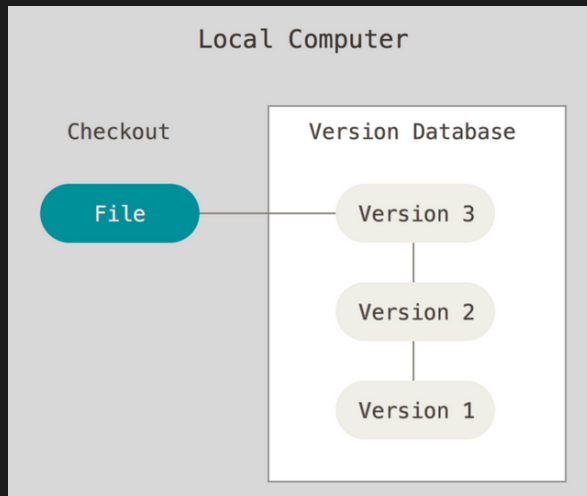
27/09/17

Version control: *software that do management of changes to documents*
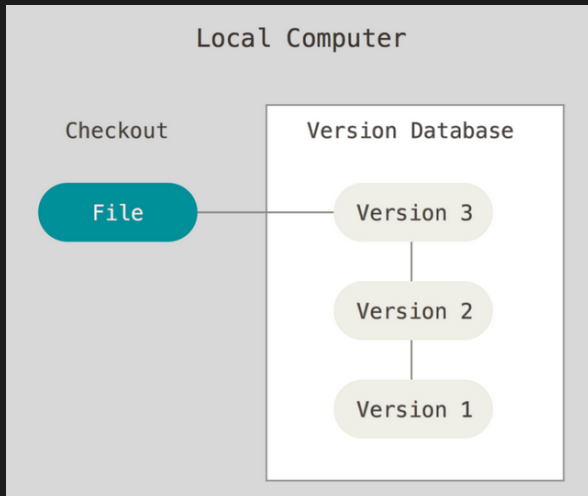
Git   Mercurial   SVN   CVS

Github   GitLab   Bitbucket   Source Forge
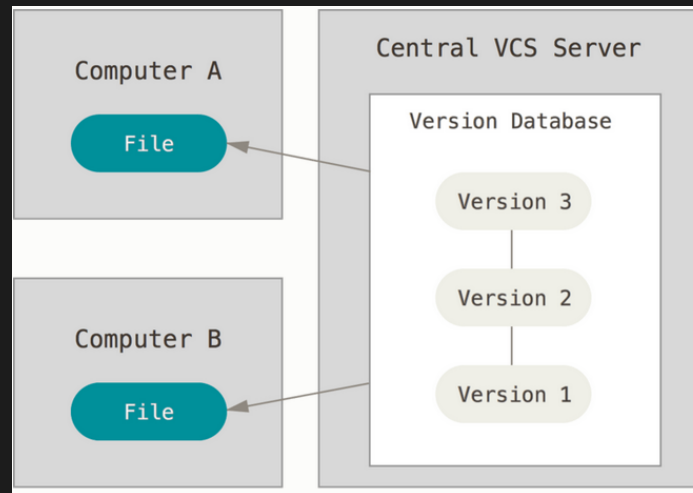
Web-based hosting service for source code.

Local version control

Local version control

Centralized version control

Credit: https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control
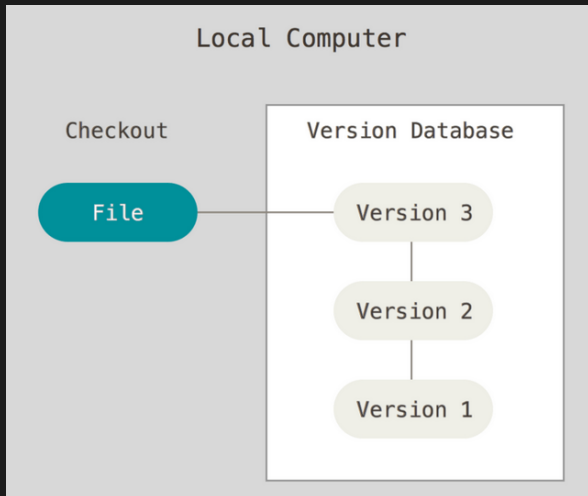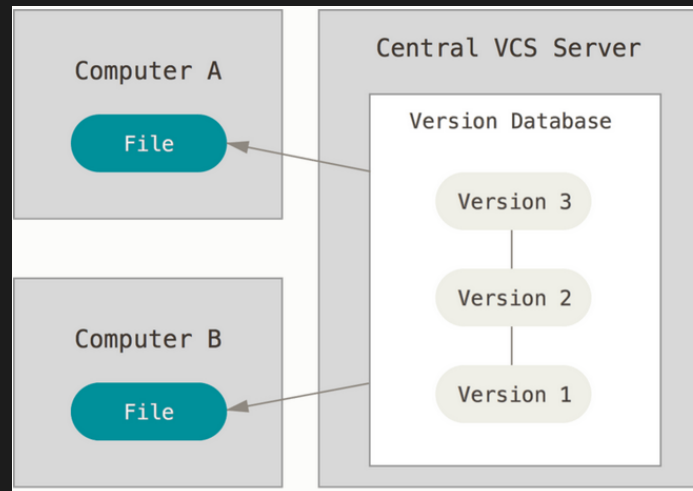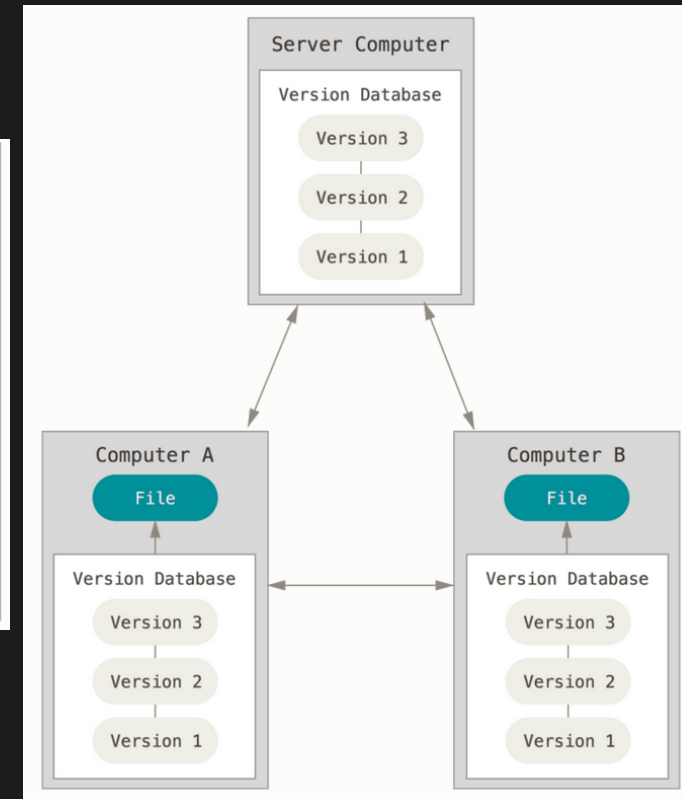
Local version control

Centralized version control

Distributed version control

Git, Mercurial, SVN, etc..
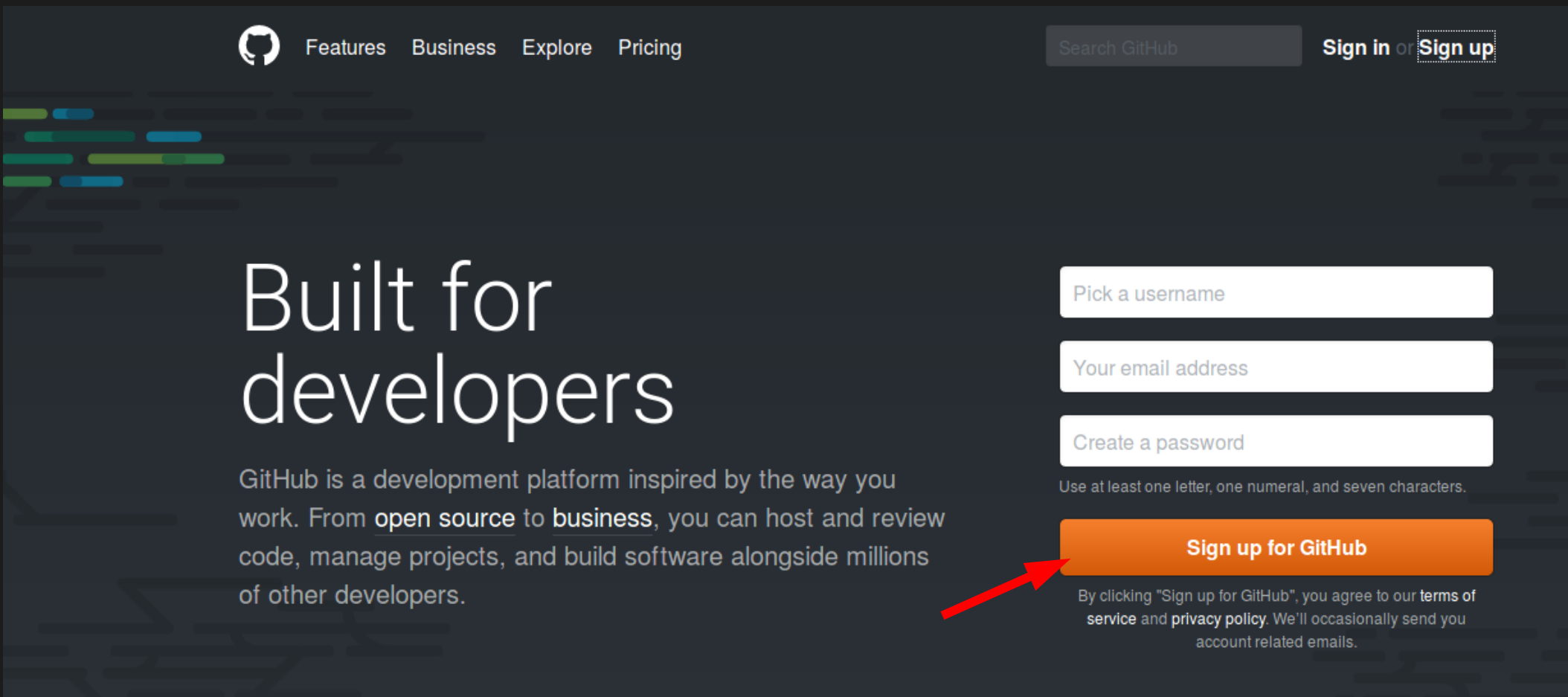
Github features

- Unlimited Public repositories, limited Private repositories.

- Documentation of software: Doc, Readme

- Issues: Report bugs or other issues with a given code.

- Email notifications.

- Github host web pages: https://pages.github.com/

- Student package: https://education.github.com/pack

Github in astronomy:

- NASA, LSST, DESI etc ..

- Astropy: https://github.com/astropy/astropy

  Matplolib, scikits-learn, emcee, and many many more.

- Help science to be reproducible.

Hands-on

1. Create a github account:

go to github.com

2. Installing git

(see the docs)

For mac: Type git and follow instructions.

For Linux:

```
$ sudo yum install git-all
$ sudo apt-get install git-all
```

For Windows:

http://git-scm.com/download/win

3. Set up git:

- Set a Git user name

```
$ git config --global user.name
"Mona Lisa"
```

- Setting your email address for every repository on your computer

```
$ git config --global user.email
"email@example.com"
```

Optional: Caching your GitHub password in Git

go here

4. Create a repository.

From github:

https://help.github.com/articles/create-a-repo/
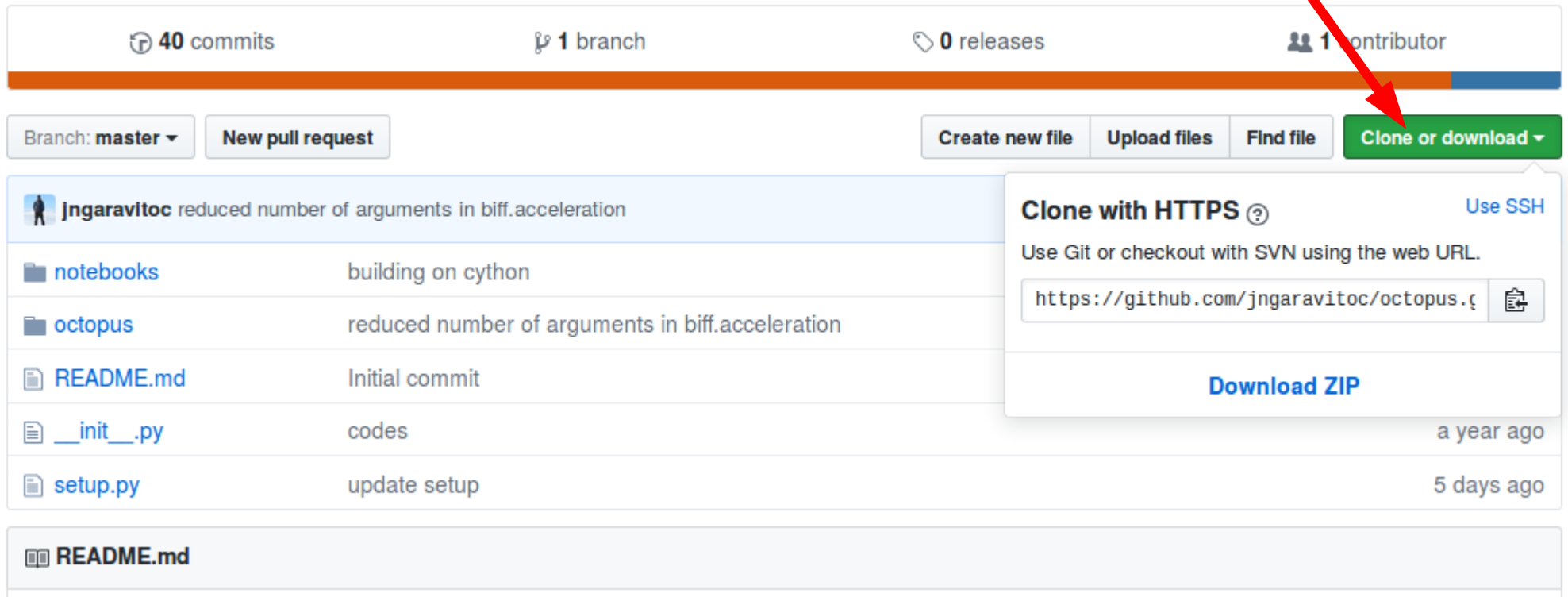
From a terminal:
Adding an existing project to github

5. Clone your repository to your computer and see its status.


```
$ cd github_repos
$ git clone https://github.com/….
$ cd your_repo_name
$ git status
```

Create a document

```
$ echo 'hello git' > git_doc.txt
```

See your repository status

```
$ git status
```

Add your file → start tracking your file (staging area)

```
$ git add first_doc.txt
$ git status
```

Commit your file → Store your file

```
$ git commit -m 'descriptive comment'
$ git log
$ git push
```

7. Editing, moving and removing files.

Move a file:

```
$ git mv file_from file_to
```

Remove a file:

```
$rm file
$ git rm file
```

Remove file from github but not from your pc.

```
$git rm --cached file
```

Do a commit through your repository web page.

```
1 lines (1 sloc)   13 Bytes                    Raw   Blame   History   ✎  🗑
```

test_orbits
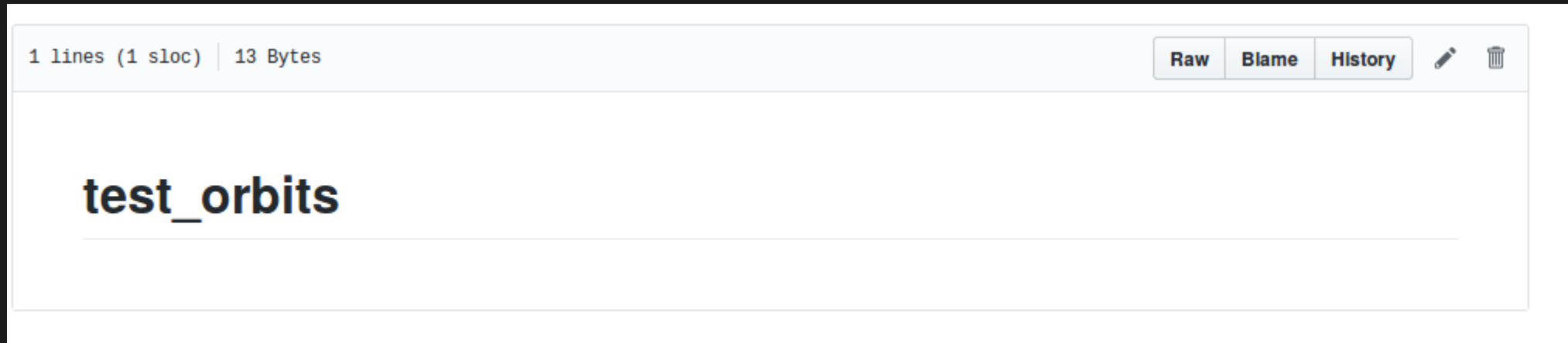
Pull your web page commits to your laptop.

On your repository type:

`$ git pull`

***Definitions*:**

Branch: Temporary version of your code in which you can do tests of a new feature.

Fork / Pull request: Your own copy of a repository, that you can edit and merge changes back (with permission), and you can receive changes from the master repository. (stackoverflow)

Clone: Local copy of a repository you can't merge back (unless you own the repository) but you can receive changes from the master repository.

8. Branches
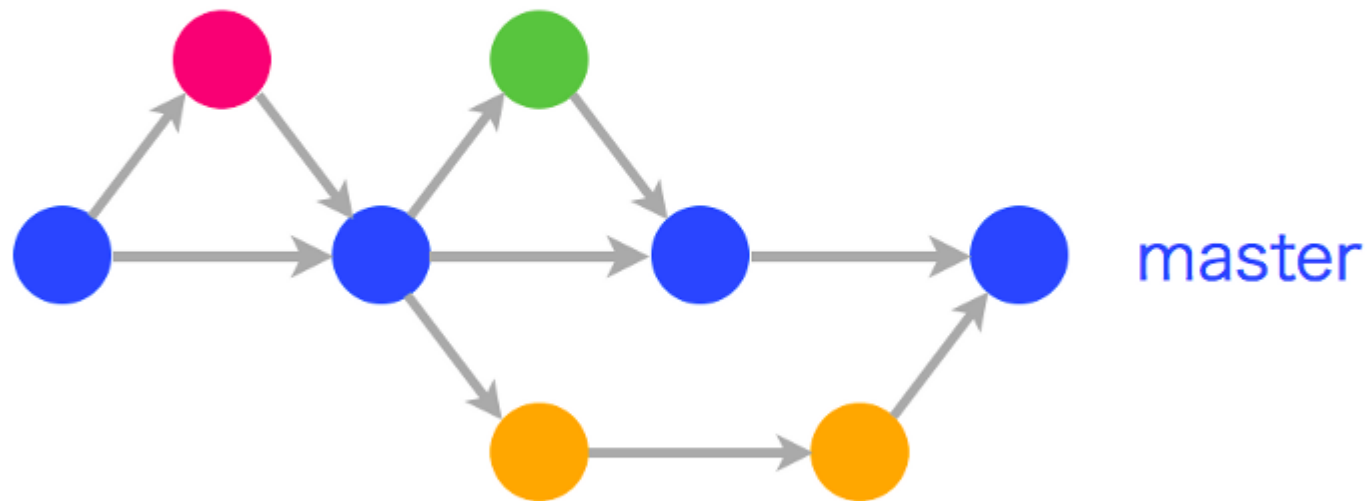
Creating a branch:

```
$ git branch branch-name   #Create new branch
$ git checkout branch-name #Move to new branch
$ git branch # Tells you which branch
```

Do some changes in the editing branch and commit those changes:

```
$ git add debugged_code.py
$ git commit -m 'fixed bug in ... '
$ git push origin branch-name
```

Merging the editing branch with the master branch:

```
$ git checkout master #Move to master branch
$ git merge branch-name # Merge branches
$ git push
$ git branch -d branch-name #Delete branch-name
$ git push origin --delete branch-name
```

9. Pull requests:

1. Create a branch.

2. Do some edits to the branch.

3. Go to your repository on github.

4. Click on New Pull Request

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: **master** ... compare: **version0.3** ✓ **Able to merge.** These branches can be automatically merged.

10. Collaborating

a) You are part of the team.

- Add a colleague to your repository.

- Clone the repository.

- Do your edits.

- Commit your edits / pull requests.

b) You are not part of the team.

- Fork or clone the repository.

- Do your edits.

- Commit your edits and make a pull request.

Good practices:

1. Document your repository.
2. Before working always: $`git pull`
3. Use descriptive comments in your commits, avoid 'update'
4. Cite and acknowledge others code in your code and repository.
5. Add a License to your repository.

Useful links

- Github help web page is very complete:

  https://help.github.com/

- Git documents are very complete and easy to read:

  https://git-scm.com/doc

- A 15 min interactive tutorial:
  https://try.github.io/levels/1/challenges/1

- Merging issues:

  used git mergetool:
  https://www.git-scm.com/docs/git-mergetool