

Lecture GIT

Objective

- I. Giới thiệu về GIT
- II. Một số khái niệm cơ bản
- III. Flow GIT Sun*
- IV. Một số Tips sử dụng



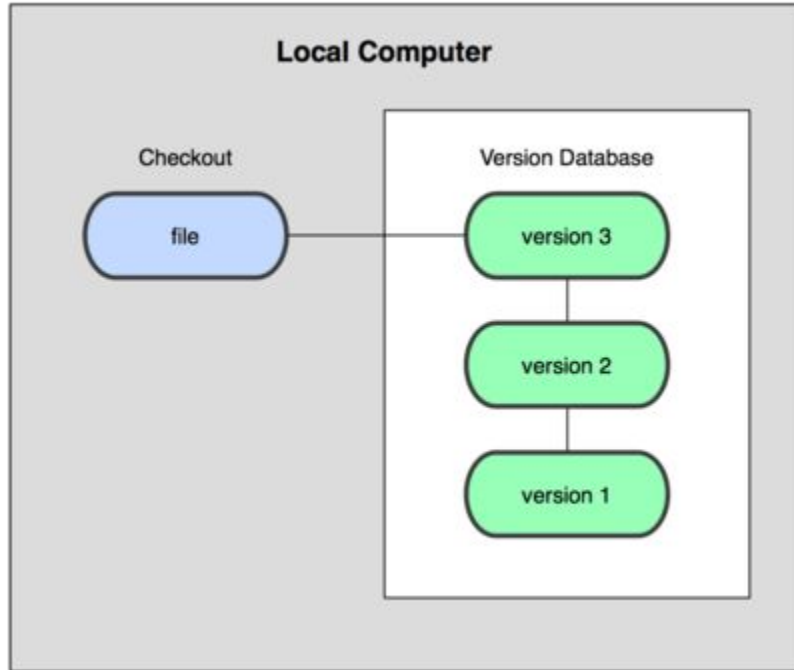
I. Giới thiệu về GIT

1. Version Control System

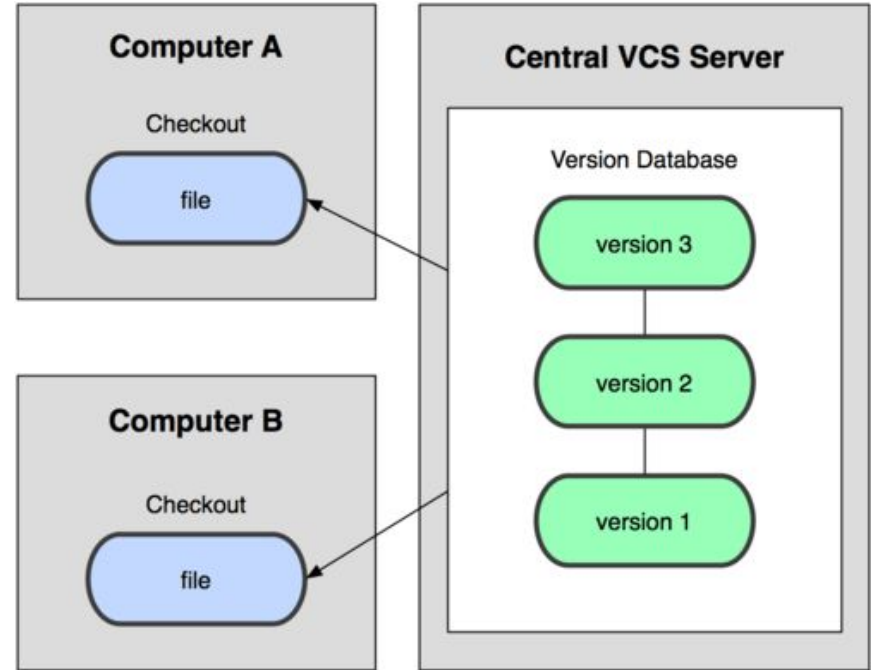
- ❖ **VCSs** là một hệ thống lưu trữ các thay đổi của một file hoặc tập hợp các file theo thời gian, do đó nó giúp bạn có thể quay lại một phiên bản xác định nào đó sau này.
- ❖ **Phân loại VCSs**
 - Hệ Thống Quản Lý Phiên Bản Cục Bộ - **Local Version Control System**
 - Hệ Thống Quản Lý Phiên Bản Tập Trung - **CVCS - Centralized Version Control System**
 - Hệ Thống Quản Lý Phiên Bản Phân Tán - **DVCS - Distributed Version Control System**

1. Version Control System

Local Version Control System

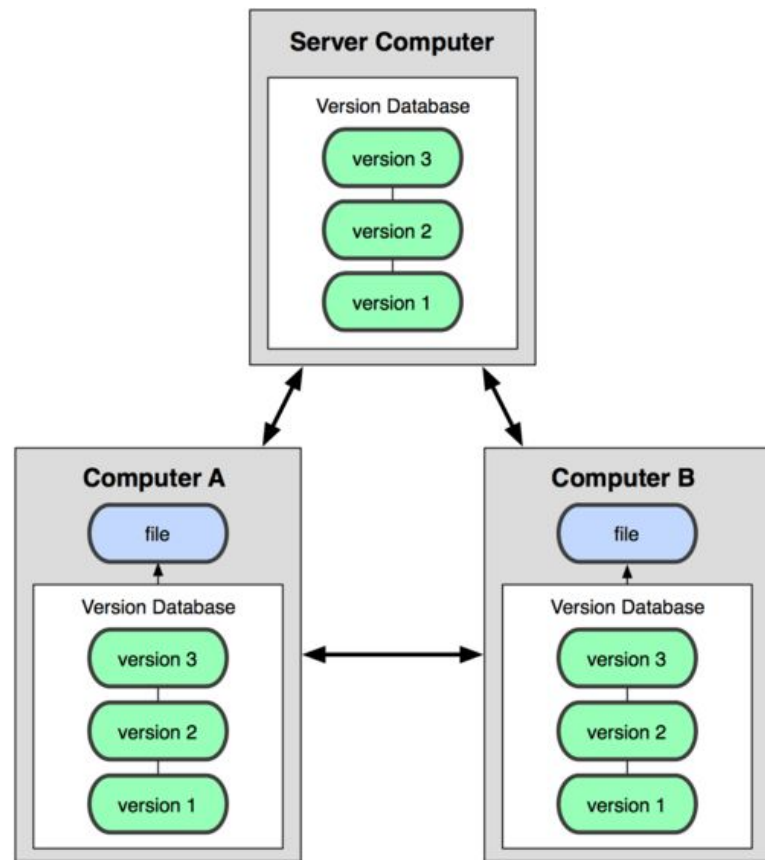
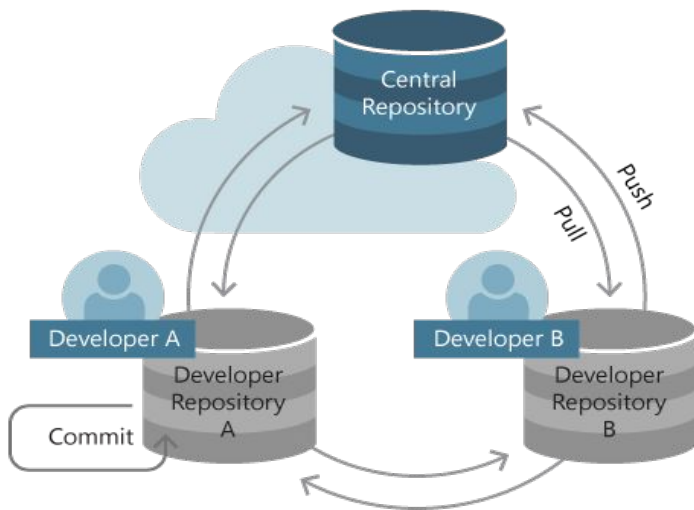


Centralized Version Control System



2. GIT là gì ?

- ❖ Git chính là một Hệ Thống Quản Lý Phiên Bản Phân Tán - **Distributed Version Control Systems (DVCSs)**



3. Ba trạng thái trong GIT



3. Ba trạng thái trong GIT

- ❖ **Modified:** Thay đổi file nhưng chưa commit vào cơ sở dữ liệu
- ❖ **Staged:** Đánh dấu sẽ commit phiên bản hiện tại của một tập tin đã chỉnh sửa trong lần commit sắp tới
- ❖ **Committed:** dữ liệu đã được lưu trữ một cách an toàn trong cơ sở dữ liệu
- ❖ **Working directory:** bản sao một phiên bản của dự án.
- ❖ **Staging area (Index):** chứa thông tin về những gì sẽ được commit trong lần commit sắp tới
- ❖ **Git directory:** nơi Git lưu trữ các metadata và cơ sở dữ liệu cho dự án của bạn.

4. Install GIT

- ❖ Xem hướng dẫn tại trang download của GIT

<https://git-scm.com/download>

- ❖ Cài đặt trên Linux sử dụng Ubuntu

```
$ apt-get install git
```

- ❖ Cấu hình danh tính của bạn

```
$ git config --global user.name "John Doe"
```

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

5. GIT help

- ❖ Nếu bạn cần sự giúp đỡ khi sử dụng Git, có ba cách để hiển thị tài liệu hướng dẫn (manpage) cho bất kỳ câu lệnh Git nào:

```
$ git help <verb>
```

```
$ git <verb> --help
```

```
$ man git-<verb>
```

- ❖ Ví dụ, bạn có thể hiển thị hướng dẫn cho câu lệnh config bằng cách chạy:

```
$ git help config
```

II. Một số khái niệm cơ bản

1. Repository

- ❖ Repository là một kho chứa, lưu trữ dữ liệu của bạn
 - Local repository: Là repository được lưu tại cục bộ (máy tính của bạn)
 - Server repository (Remote repository): Là repository nhưng được lưu tại server của các hosting-service sử dụng Git (ví dụ: **Github**, **GitLab**, **Bitbucket** ...)
- ❖ Tạo một git repository:

```
$ git init
```


- ❖ Sao chép Repository đã tồn tại:

```
$ git clone [url]
```

```
ví dụ: git clone https://github.com/schacon/grit.git
```






1. Repository

 **schacon / grit**
forked from [mojombo/grit](#)

 Watch **3**  Star **164**  Fork **485**






[Code](#) [Issues 0](#) [Pull requests 1](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#)


Grit is a Ruby library for extracting information from a git repository in an object oriented manner - this fork tries to intergrate as much pure-ruby functionality as possible <http://grit.rubyforge.org/>

 **333** commits  **3** branches  **1** release  **23** contributors  MIT

Branch: **master** [New pull request](#) [Create new file](#) [Upload files](#) [Find File](#) [Clone or download](#)

This branch is 183 commits behind [mojombo:master](#).

| | |
|--|--|
|  mojombo implement select_existing_objects | |
|  examples | added examples |
|  lib | implement select_existing_objects |
|  test | implement select_existing_objects |
|  .gitignore | Added Repo#commit_deltas_from as a (fairly expensive and lazy) way of... |

Clone with HTTPS [Use SSH](#)
Use Git or checkout with SVN using the web URL.
`https://github.com/schacon/grit.git` 
[Download ZIP](#)

9 years ago
11 years ago

URL using clone

2. Commit

- ❖ Commit: ghi lại Thay Đổi vào git repository. một version mới được tạo ra bằng cách tạo một "commit" cho các sự thay đổi của dữ liệu.
- ❖ Tạo mới một commit:

```
$ git commit -m "Nội dung commit"
```
- ❖ Lưu lại thay đổi nhưng ghi đè lên commit trước đó:

```
$ git commit --amend -m "Nội dung mới nếu có thay đổi"
```
- ❖ Xem lại danh sách commit đã tạo, mỗi commit trên một dòng:

```
$ git log --oneline
```

2. Commit

- ❖ Kiểm tra trạng thái của file:

```
$ git status  
# On branch master  
nothing to commit, working directory clean
```

- ❖ Theo Dõi Các Tập file mới:

```
$ git add <file_name>
```

- ❖ Xem thay đổi giữa working với stage:

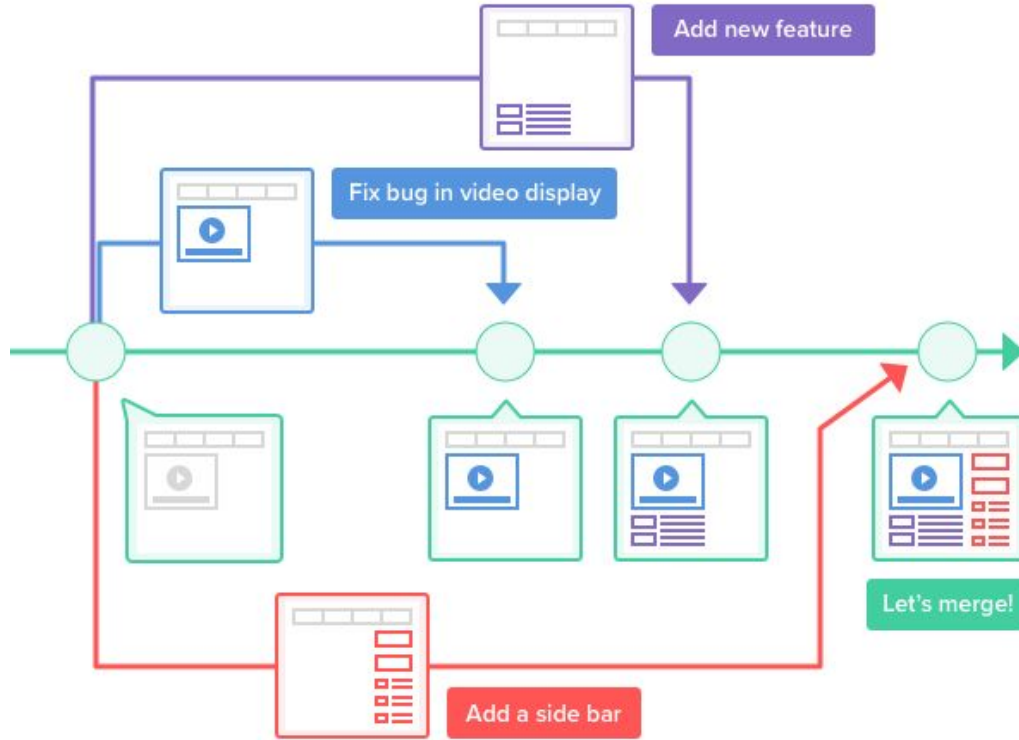
```
$ git diff
```

2. Commit

- ❖ Bỏ qua các file: đưa danh sách các file không muốn theo dõi `.gitignore`
- ❖ Đây là một ví dụ của `.gitignore`:

```
# không theo dõi tập tin có đuôi .a
*.a
# nhưng theo dõi tập lib.a, mặc dù bạn đang bỏ qua tất cả tập tin .a ở trên
!lib.a
# chỉ bỏ qua tập TODO ở thư mục gốc, chứ không phải ở các thư mục con
/TODO
# bỏ qua tất cả tập tin trong thư mục build/
build/
# bỏ qua doc/notes.txt, không phải doc/server/arch.txt
doc/*.txt
# bỏ qua tất cả tập .txt trong thư mục doc/
doc/**/*txt
```


3. Branch



3. Branch

- ❖ Branch: Một nhánh trong Git là một con trỏ (index) có khả năng di chuyển được, trỏ đến một trong những commit.
- ❖ Tạo mới branch:

```
$ git branch <branch_name>
```

```
vd: git branch testing
```

- ❖ Để chuyển sang một nhánh đang tồn tại:

```
$ git checkout <branch_name>
```

```
vd: git checkout master
```

3. Branch

- ❖ Để tạo một nhánh và chuyển sang nhánh đó đồng thời:

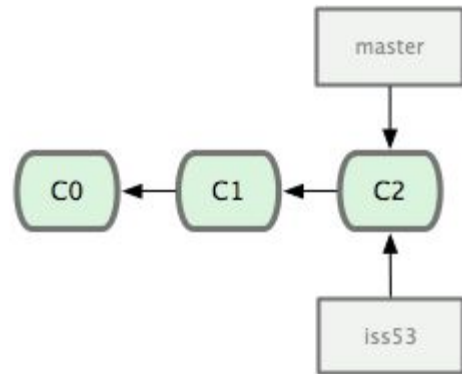
```
$ git checkout -b <branch_name>
```

```
vd: git checkout -b iss53
```

- ❖ Xóa một nhánh:

```
$ git branch -d <branch_name>
```

```
vd: git branch -d iss53
```



3. Branch

- ❖ Để tạo một nhánh và chuyển sang nhánh đó đồng thời:

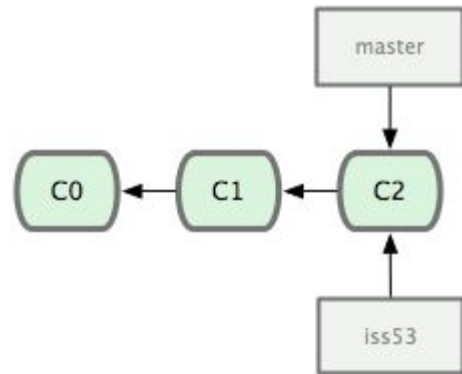
```
$ git checkout -b <branch_name>
```

```
vd: git checkout -b iss53
```

- ❖ Xóa một nhánh:

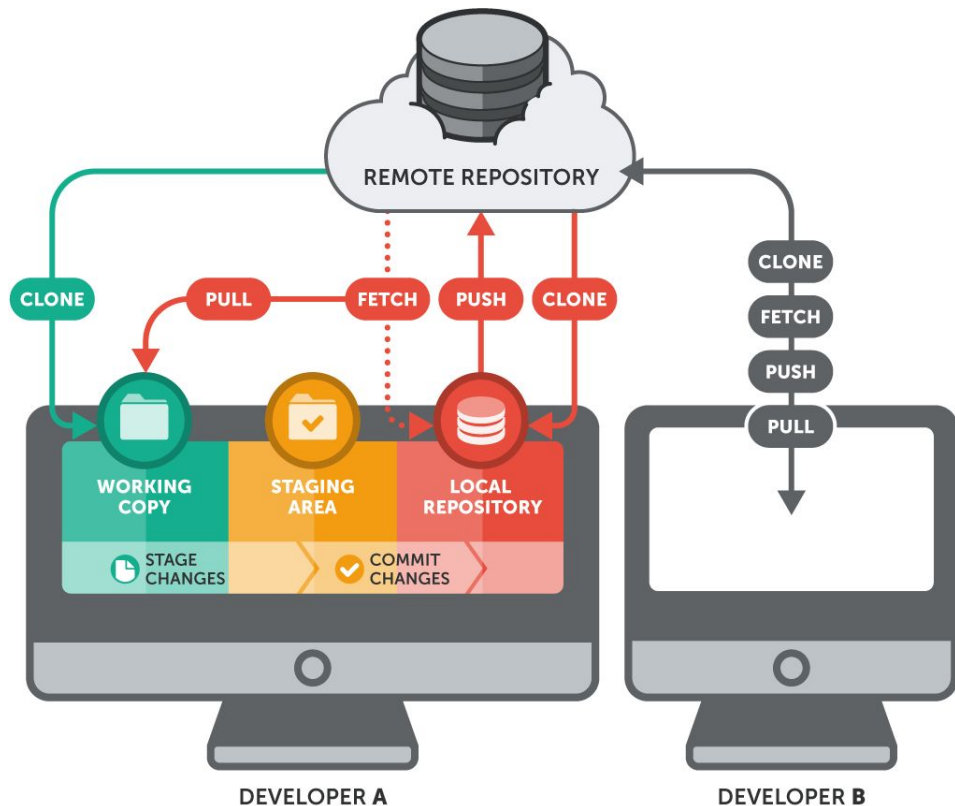
```
$ git branch -d <branch_name>
```

```
vd: git branch -d iss53
```



4. Remote

- ❖ Mỗi tham chiếu đến server repository được gọi là một remote
- ❖ Mỗi remote sẽ có các thông tin:
 - Tên remote
 - URL: đường link của server repository (cung cấp 2 giao thức http hoặc ssh)



4. Remote

❖ Thêm một remote:

```
$ git remote add <remote_name> <url>
```

```
vd: git remote add sun-asterisk
```

```
git@github.com:framgia/git_lecture.git
```

4. Remote

- ❖ Xem danh sách remote:

```
$ git remote -v
```

- ❖ Thay đổi url của remote:

```
$ git remote set-url <remote_name> <remote_url>
```

```
vd: git remote set-url sun-asterisk
```

```
https://github.com/framgia/git_lecture.git
```

4. Remote

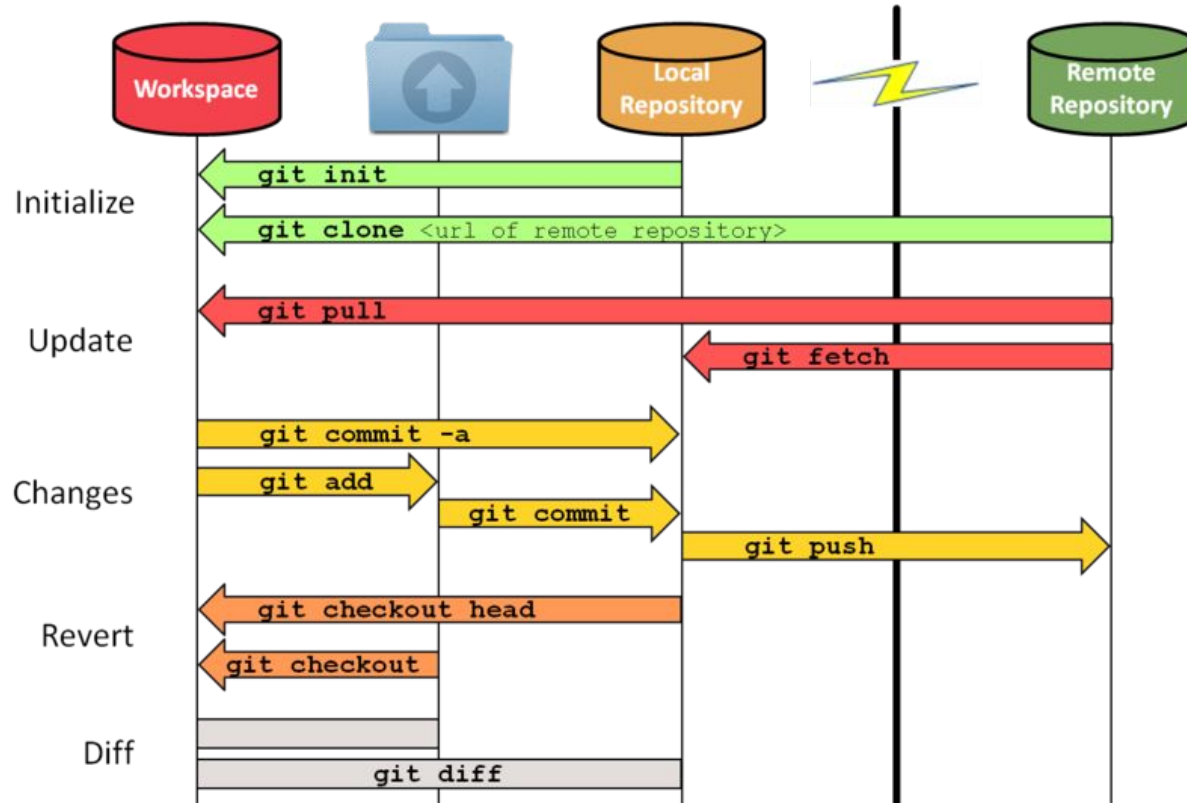
- ❖ Thay đổi tên của remote:

```
$ git remote rename <old_name> <new_name>  
vd: git remote rename sun-asterisk framgia
```

- ❖ Xóa một remote:

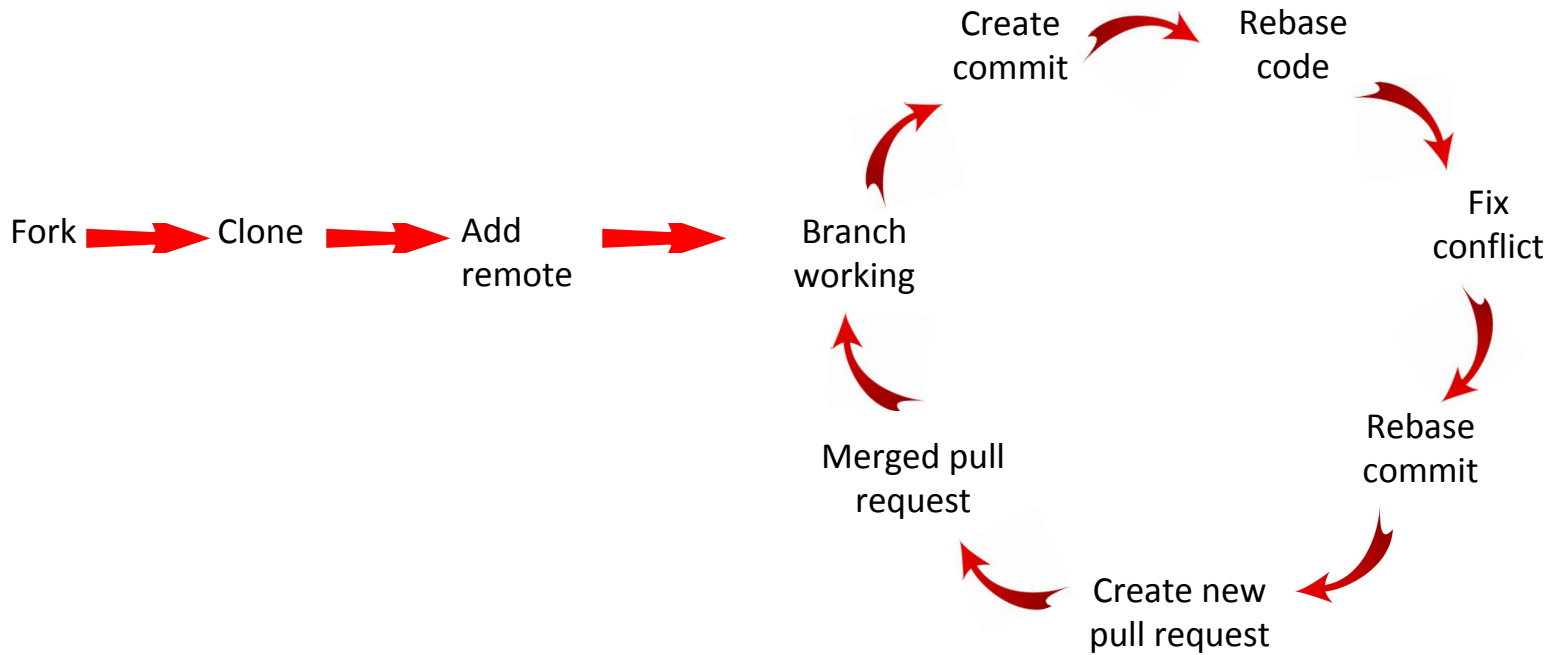
```
$ git remote remove <remote_name>  
vd: git remote remove framgia
```


4. Remote



III. GIT Flow

1. GIT Flow



Step 1: Fork

jump to... / Pull requests Issues Marketplace Explore

framgia / git_lecture

Watch 22 Star 0 Fork 3

Code Issues 0 Pull requests 8 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided.

Manage topics

17 commits 2 branches 0 releases 3 contributors

Branch: master New pull request Create new file Upload files Find File Clone or download

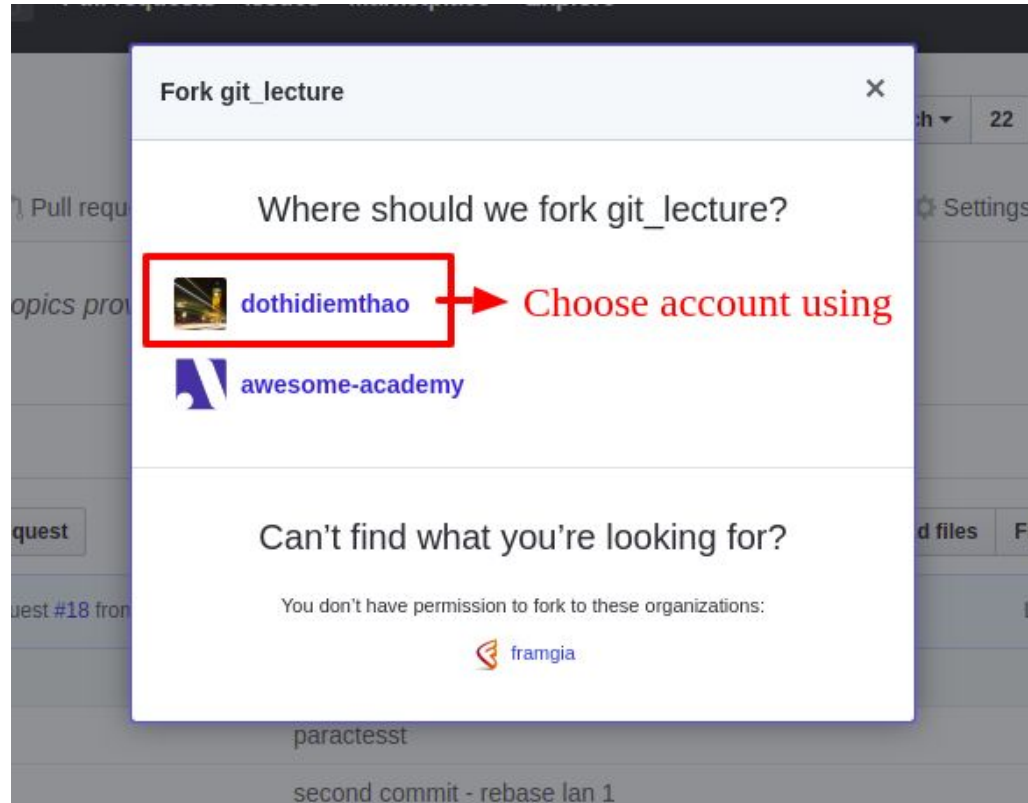
VuThiTranVan Merge pull request #18 from VuThiTranVan/demo-24-02 Latest commit 79d8b6a on Apr 24

| | | |
|-------------------------------------|------------------------------|--------------|
| .gitignore | commit lan 1 + rebase HEAD~2 | 3 months ago |
| aa.txt | paractesst | 2 months ago |
| demo-batch23-01.txt | second commit - rebase lan 1 | 2 months ago |

Click here

Edit

Step 1: Fork



Step 1: Fork

The screenshot shows a GitHub repository page for 'dothidiemthao / git_lecture', which is a fork of 'framgia/git_lecture'. The repository has 0 stars, 0 forks, and 3 forks. The 'Code' tab is selected, showing the repository's structure. The repository has 17 commits, 2 branches, and 0 releases. The 'master' branch is selected, and the 'Clone or download' button is highlighted in green. The repository contains files: .gitignore, aa.txt, demo-batch23-01.txt, and demo-batch23-02.txt. The repository is currently fetching the latest commit.

dothidiemthao / **git_lecture**
forked from framgia/git_lecture

Watch 0 Star 0 Fork 3

Code Pull requests 0 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. [Edit](#)

[Manage topics](#)

17 commits 2 branches 0 releases Fetching contributors

Branch: master New pull request Create new file Upload files Find File Clone or download

This branch is even with framgia:master. [Pull request](#) [Compare](#)

Fetching latest commit...

- .gitignore
- aa.txt
- demo-batch23-01.txt
- demo-batch23-02.txt

com...

Step 2: Clone

The screenshot shows the GitHub interface for the repository 'dothidiemthao / git_lecture', which was forked from 'framgia/git_lecture'. The repository has 17 commits, 2 branches, 0 releases, and 3 contributors. The 'Code' tab is selected, showing a list of files: .gitignore, aa.txt, demo-batch23-01.txt, and demo-batch23-02.txt. A red box highlights the 'Clone or download' button, with an arrow pointing to it labeled 'step 1.'. A dropdown menu is open, showing 'Clone with SSH' (highlighted with a red box) and 'Use HTTPS'. The SSH URL 'git@github.com:dothidiemthao/git_lecture' is displayed, with a red box around the copy icon and an arrow pointing to it labeled 'step 2.'. The 'Download ZIP' option is also visible.

dothidiemthao / git_lecture
forked from framgia/git_lecture

Watch 0 Star 0 Fork 3

Code Pull requests 0 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. [Manage topics](#)

17 commits 2 branches 0 releases 3 contributors

Branch: master New pull request

Create new file Upload files Find File Clone or download

This branch is even with framgia:master.

VuThiTranVan Merge pull request framgia#18 from VuThiTranVan/demo-24-02

| | |
|---------------------|------------------------------|
| .gitignore | commit lan 1 + rebase HEAD~2 |
| aa.txt | paracetsst |
| demo-batch23-01.txt | second commit - rebase lan 1 |
| demo-batch23-02.txt | second commit - rebase lan 1 |

Step 2: Clone

- ❖ Cd đến thư mục muốn lưu trữ repo cần sao chép, ví dụ ở đây là project

```
$ cd project
```

- ❖ Thực hiện clone repository tương ứng với url đã copy, sử dụng giao thức SSH

```
$ git clone
```

```
git@github.com:dothidiemthao/git_lecture.git
```

```
→ project git clone git@github.com:dothidiemthao/git_lecture.git
Cloning into 'git_lecture'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 46 (delta 2), reused 19 (delta 2), pack-reused 26
Receiving objects: 100% (46/46), 9.86 KiB | 0 bytes/s, done.
Resolving deltas: 100% (7/7), done.
Checking connectivity... done.
```


Step 3: Add remote

❖ Copy url repository đã fork

The screenshot shows the GitHub interface for the repository 'framgia / git_lecture'. The repository name is highlighted with a red box. The repository has 22 watches, 0 stars, and 3 forks. The main content area shows a list of files and commits, with a pull request #18 from VuThiTranVan. A red box highlights the 'Clone or download' button. A modal window is open showing the 'Clone with SSH' option, with the SSH URL 'git@github.com:framgia/git_lecture.git' and a red box around the copy icon.

framgia / git_lecture

Watch 22 Star 0 Fork 3

Code Issues 0 Pull requests 8 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. Edit

Manage topics

17 commits 2 branches 0 releases 3 contributors

Branch: master New pull request Create new file Upload files Find File Clone or download

VuThiTranVan Merge pull request #18 from VuThiTranVan/demo-24-02

| | |
|---------------------|------------------------------|
| .gitignore | commit lan 1 + rebase HEAD~2 |
| aa.txt | paractesst |
| demo-batch23-01.txt | second commit - rebase lan 1 |
| demo-batch23-02.txt | second commit - rebase lan 1 |
| tesst.txt | second commit - rebase lan 1 |

Clone with SSH Use HTTPS

Use an SSH key and passphrase from account.

git@github.com:framgia/git_lecture.git

Download ZIP

Step 3: Add remote

- ❖ Add remote repository với tên tùy chọn, ví dụ ở đây là sun-asterisk sử dụng giao thức SSH

```
$ git remote add sun-asterisk  
git@github.com:framgia/git_lecture.git  
$ git remote -v #show list remote
```

```
→ git_lecture git:(master) git remote add sun-asterisk git@github.com:framgia/git_lecture.git  
→ git_lecture git:(master) git remote -v  
origin git@github.com:dothidiemthao/git_lecture.git (fetch)  
origin git@github.com:dothidiemthao/git_lecture.git (push)  
sun-asterisk git@github.com:framgia/git_lecture.git (fetch)  
sun-asterisk git@github.com:framgia/git_lecture.git (push)  
→ git_lecture git:(master) █
```

Step 4, 5: Create new branch, Commit data

- ❖ Checkout branch mới từ branch master

```
$ git checkout -b add_new
```

- ❖ Sửa đổi dữ liệu, add thay đổi và commit thay đổi

```
→ git_lecture git:(add_new) git status
On branch add_new
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    new.txt

nothing added to commit but untracked files present (use "git add" to track)
→ git_lecture git:(add_new) X git add .
→ git_lecture git:(add_new) X git commit -m "First commit"
[add_new f6e5b66] First commit
 1 file changed, 1 insertion(+)
 create mode 100644 new.txt
→ git_lecture git:(add_new) █
```

Step 6: Push code

- ❖ Checkout branch mới từ branch master

```
$ git push <remote_name> <local_branch>:<repo_branch>
```

```
→ git_lecture git:(add_new) git push origin add_new:add_new
Counting objects: 8, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 398 bytes | 0 bytes/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'add_new' on GitHub by visiting:
remote:   https://github.com/dothidiemthao/git_lecture/pull/new/add_new
remote:
To git@github.com:dothidiemthao/git_lecture.git
 * [new branch]      add_new -> add_new
→ git_lecture git:(add_new) █
```

Step 7: Create pull request

❖ Cách 1:

The screenshot shows the GitHub interface for the repository 'framgia / git_lecture'. The repository name is highlighted with a red box. The navigation bar includes links for Code, Issues (0), Pull requests (8), Projects (0), Wiki, Security, Insights, and Settings. A description placeholder states 'No description, website, or topics provided.' with an 'Edit' button and a 'Manage topics' link.

Repository statistics are displayed: 19 commits, 2 branches, 0 releases, and 3 contributors.

Under 'Your recently pushed branches:', a branch 'dothidiemthao:add_new_more' (pushed 2 minutes ago) is highlighted with a red box. To its right, a green button labeled 'Compare & pull request' is also highlighted with a red box.

Below the branches section, the 'Branch: master' dropdown is shown next to a 'New pull request' button. To the right are buttons for 'Create new file', 'Upload files', 'Find File', and a green 'Clone or download' button.

The commit history shows a merge by 'dothidiemthao' of pull request #19 from the 'add_new' branch, with the latest commit '09ec6dc' made 3 minutes ago. Below this, a commit for '.gitignore' is listed as the 'First commit' made 5 minutes ago.

Step 7: Create pull request

❖ Cách 2:

framgia / git_lecture

Watch 22 Star 0 Fork 3

Code Issues 0 Pull requests 8 Projects 0 Wiki Security Insights Settings

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 repository: framgia/git_lecture base: master head repository: dothidiemthao/git_lecture compare: add_new_more

✗ Can't automatically merge. Don't worry, you can still create the pull request.

Create pull request Discuss and review the changes in this comparison with others.



1 commit 2 files changed 0 commit comments 1 contributor





Commits on Jun 10, 2019



dothidiemthao Add new and fix d5e6f5a

Step 8: Fix conflict



Add new and fix #20

 **Open** dothidiemthao wants to merge 1 commit into `framgia:master` from `dothidiemthao:add_new_more` 



 Conversation **0**  Commits **1**  Checks **0**  Files changed **2**

 dothidiemthao commented in 30 seconds Member +  ...

No description provided.

  Add new and fix d5e6f5a


Add more commits by pushing to the `add_new_more` branch on `dothidiemthao/git_lecture`.

  **This branch has conflicts that must be resolved** Resolve conflicts

Use the [web editor](#) or the [command line](#) to resolve conflicts.

Conflicting files

`.gitignore`

Merge pull request  or view [command line instructions](#).

Step 8: Fix conflict

```
→ git_lecture git:(add_new_more) git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
→ git_lecture git:(master) git pull sun-asterisk master
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), done.
From github.com:frangia/git_lecture
 * branch          master      -> FETCH_HEAD
 * [new branch]     master      -> sun-asterisk/master
Updating 79d8b0a..09ec6dc
Fast-forward
 .gitignore | 2 +-
 new.txt    | 1 +
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 new.txt
→ git_lecture git:(master)
```


Step 8: Fix conflict

```
→ git_lecture git:(master) git checkout add_new_more
Switched to branch 'add_new_more'
→ git_lecture git:(add_new_more) git rebase master
First, rewinding head to replay your work on top of it...
Applying: Add new and fix
Using index info to reconstruct a base tree...
M    .gitignore
Falling back to patching base and 3-way merge...
Auto-merging .gitignore
CONFLICT (content): Merge conflict in .gitignore
Failed to merge in the changes.
Patch failed at 0001 Add new and fix
The copy of the patch that failed is found in:
    /home/frangia/project/git_lecture/.git/rebase-apply/patch

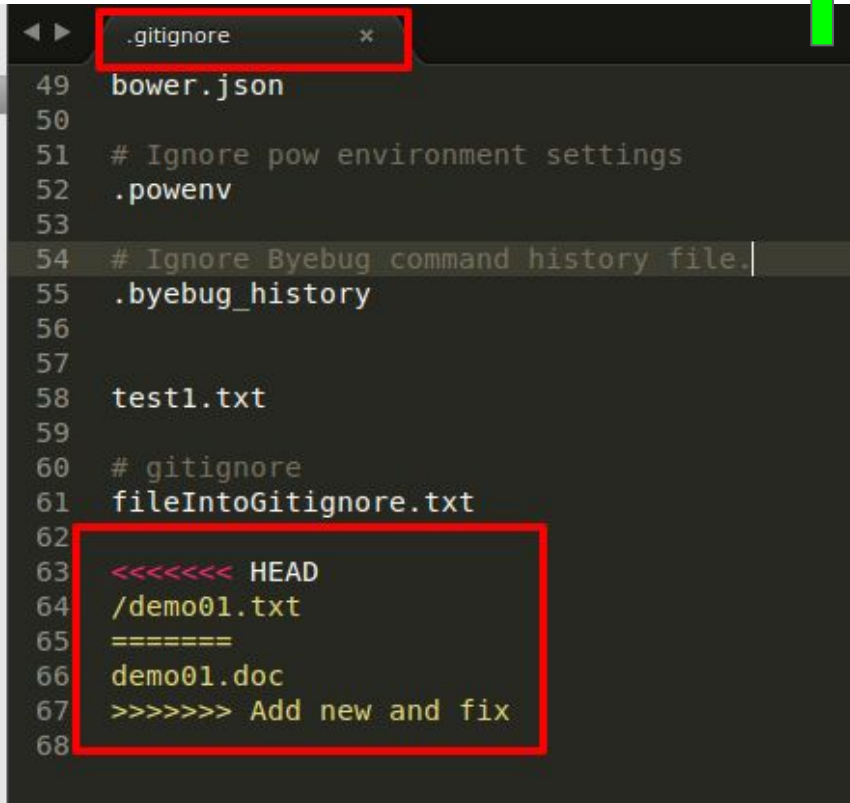
When you have resolved this problem, run "git rebase --continue".
If you prefer to skip this patch, run "git rebase --skip" instead.
To check out the original branch and stop rebasing, run "git rebase --abort".
```

Step 8: Fix conflict

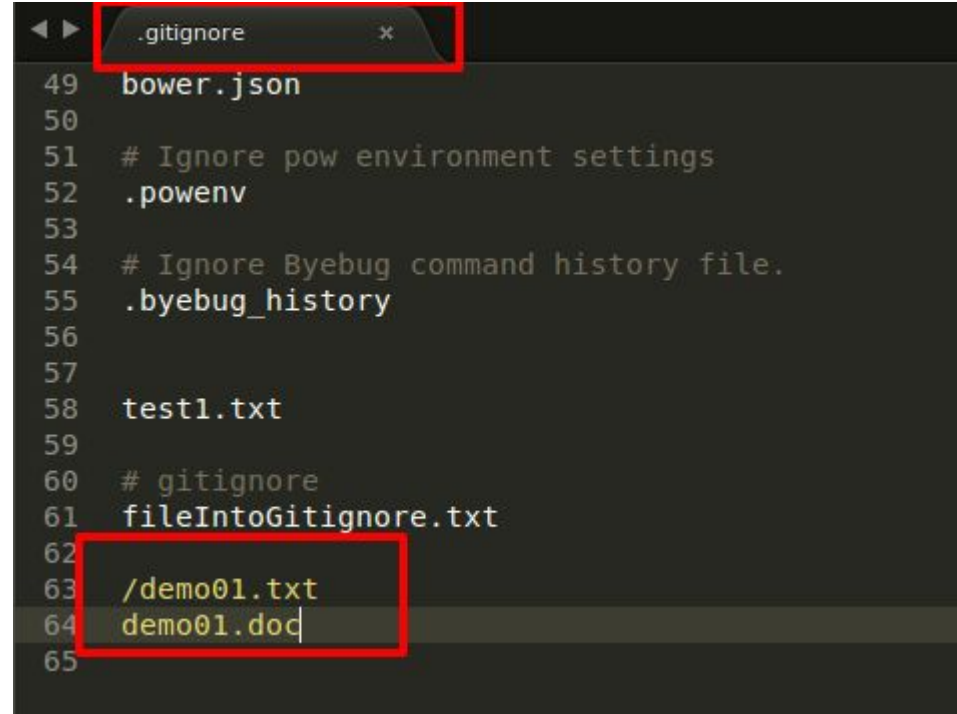
```
→ git_lecture git:(master) git checkout add_new_more
Switched to branch 'add_new_more'
→ git_lecture git:(add_new_more) git rebase master
First, rewinding head to replay your work on top of it...
Applying: Add new and fix
Using index info to reconstruct a base tree...
M    .gitignore
Falling back to patching base and 3-way merge...
Auto-merging .gitignore
CONFLICT (content): Merge conflict in .gitignore
Failed to merge in the changes.
Patch failed at 0001 Add new and fix
The copy of the patch that failed is found in:
    /home/frangia/project/git_lecture/.git/rebase-apply/patch

When you have resolved this problem, run "git rebase --continue".
If you prefer to skip this patch, run "git rebase --skip" instead.
To check out the original branch and stop rebasing, run "git rebase --abort".
```

Step 8: Fix conflict



```
49 bower.json
50
51 # Ignore pow environment settings
52 .powenv
53
54 # Ignore Byebug command history file.
55 .byebug_history
56
57 test1.txt
58
59
60 # gitignore
61 fileIntoGitignore.txt
62
63 <<<<<<< HEAD
64 /demo01.txt
65 =====
66 demo01.doc
67 >>>>>>> Add new and fix
68
```



```
49 bower.json
50
51 # Ignore pow environment settings
52 .powenv
53
54 # Ignore Byebug command history file.
55 .byebug_history
56
57 test1.txt
58
59
60 # gitignore
61 fileIntoGitignore.txt
62
63 /demo01.txt
64 demo01.doc
65
```

Step 8: Fix conflict

```
→ git_lecture git:(09ec6dc) ✗ git status
rebase in progress; onto 09ec6dc
You are currently rebasing branch 'add_new_more' on '09ec6dc'.
  (fix conflicts and then run "git rebase --continue")
  (use "git rebase --skip" to skip this patch)
  (use "git rebase --abort" to check out the original branch)

Unmerged paths:
  (use "git reset HEAD <file>..." to unstage)
  (use "git add <file>..." to mark resolution)

        both modified:   .gitignore

no changes added to commit (use "git add" and/or "git commit -a")
→ git_lecture git:(09ec6dc) ✗ git add .
→ git_lecture git:(09ec6dc) ✗ git rebase --continue
Applying: Add new and fix
→ git_lecture git:(add_new_more) █
```

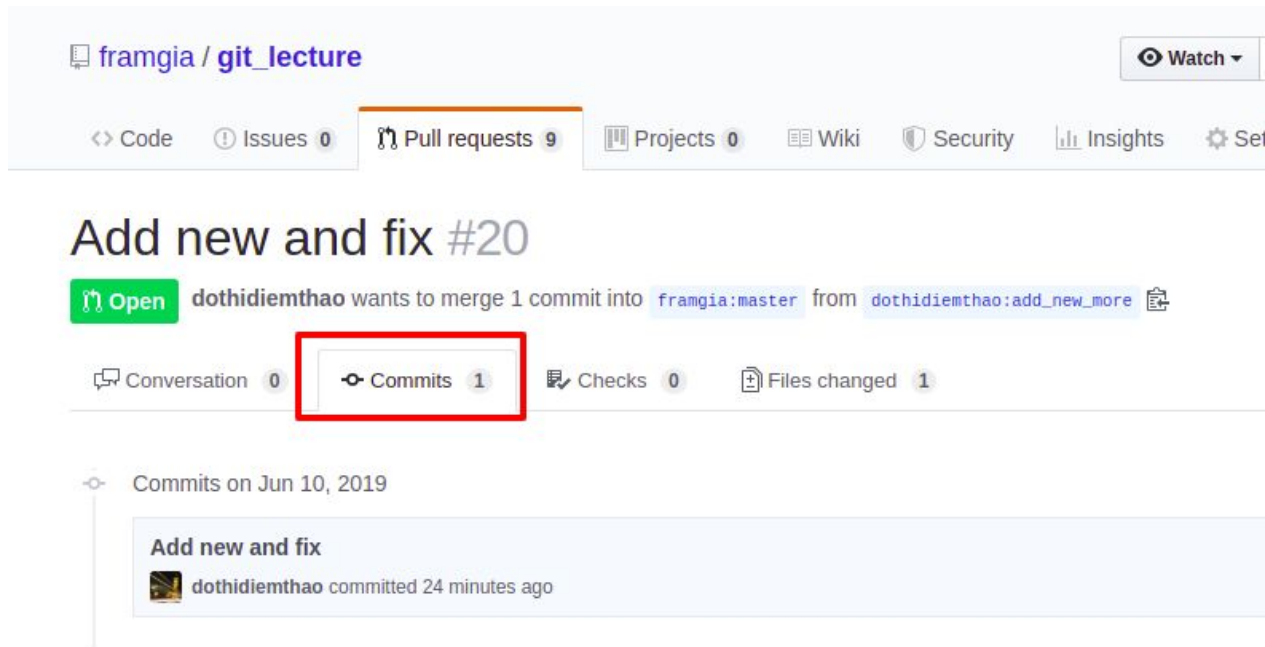

Step 8: Fix conflict

```
→ git_lecture git:(add_new_more) git push origin add_new_more -f
Counting objects: 12, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 968 bytes | 0 bytes/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To git@github.com:dothidiemthao/git_lecture.git
+ d5e6f5a...db09b37 add_new_more -> add_new_more (forced update)
→ git_lecture git:(add_new_more) █
```

Một số lưu ý khi tạo pull request

- ❖ Số lượng commit nên là:



1 commit / pull








Một số lưu ý khi tạo pull request

- ❖ Check pull không bị conflict


Add new and fix #20


 **Open** dothidiemthao wants to merge 1 commit into `framgia:master` from `dothidiemthao:add_new_more` 

 Conversation **0**  Commits **1**  Checks **0**  Files changed **1**



 dothidiemthao commented 10 minutes ago Member + 😊 ...


No description provided.

 Add new and fix db09b37

 dothidiemthao force-pushed the `dothidiemthao:add_new_more` branch from `d5e6f5a` to `db09b37` now

Add more commits by pushing to the `add_new_more` branch on `dothidiemthao/git_lecture`.

  **Continuous integration has not been set up**
Several apps are available to automatically catch bugs and enforce style.

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

IV. Một số tips sử dụng

1. Gộp nhiều commit thành một commit

```
$ git rebase -i HEAD~<số lượng commit>
# Ví dụ) git rebase -i HEAD~3
# (trước khi sửa) các commit cũ từ trên xuống dưới
pick aa11bbc commit message 1
pick b2c3c4d commit message 2
pick 4e56fgh commit message 3
. . .
```

1. Gộp nhiều commit thành một commit

(sau khi sửa) các commit trước là squash sẽ được tổng hợp vào 1 commit ở đây trước

pick aallbbc commit message 1

squash b2c3c4d commit message 2

squash 4e56fgh commit message 3

...

Thay vì dùng squash có thể dùng fixup nhưng khi đó commit message của những commit được fixup sẽ bị mất

2. Sửa nội dung của commit cũ

```
$ git rebase -i <commit_id>
# (trước khi sửa) các commit cũ từ trên xuống dưới
pick aallbbc commit message 1
pick b2c3c4d commit message 2

# (sau khi sửa) Đổi commit cần sửa sang edit
edit aallbbc commit message 1
pick b2c3c4d commit message 2

$ git commit --amend -m "Nội dung commit mới cần đổi"
$ git rebase --continue
```

3. Ignore một số file commit thừa

```
# Đầu tiên là xoá các file đã commit khỏi repository  
$ git rm --cached <tên file>
```

```
# Sau đó là thêm vào gitignore
```

```
$ echo '<tên file>' >> .gitignore
```

4. Đổi tên branch cũ

```
$ git branch -m <tên branch sau khi đổi>
```

5. Commit nhầm sang một branch khác

Đầu tiên là tạo một branch khác chứa trạng thái mà ta đã commit

```
$ git branch other-branch
```

Đưa HEAD, index của master về 1 commit trước đó

```
$ git reset --hard HEAD~
```

Check out sang branch có commit trước đó

```
$ git checkout other-branch
```

6. Lỡ tay commit và muốn xóa commit

1. Chỉ đưa HEAD về như cũ

```
$ git reset --soft HEAD~
```

2. Đưa HEAD và index về như cũ

```
$ git reset HEAD~
```

3. Đưa cả index, working tree về 1 commit trước đó

```
$ git reset --hard HEAD~
```

4. Xóa luôn commit, về lại trước đó

```
$ git revert <commit>
```

7. Đang làm dở dang và chuyển branch khác

```
# Tạm thời lưu lại các phần công việc còn đang làm dở
$ git stash -u

# Chuyển sang một branch khác và làm việc
$ git checkout -b other-branch

~làm việc, làm việc, làm việc~

$ git add <các file cần thiết>
$ git commit -m "commit message"

# Trở về branch cũ
$ git checkout origin-branch

# Lấy lại các nội dung công việc đang làm dở trước đó
$ git stash pop
```

8. Lỡ xóa mất commit quan trọng

Đầu tiên là xem lại toàn bộ lịch sử commit

```
$ git reflog
```

Từ đó chọn commit muốn phục hồi và khôi phục lại

```
$ git reset --hard <commit>
```

9. Lỡ xóa mất branch

Đầu tiên là xem lại toàn bộ lịch sử commit

```
$ git reflog
```

Từ các commit này, chọn rồi tạo branch mới

ví dụ) git branch new-branch HEAD@{2}

```
$ git branch <tên branch> <commit>
```


10. Đã merge nhưng muốn trở lại như trước

```
# tiến hành merge
```

```
$ git checkout <tên branch nguồn>
```

```
$ git merge <tên branch muốn merge>
```

```
# Sau khi merge, nhưng lại muốn trở lại như trước thì làm như sau
```

```
$ git reset --hard ORIG_HEAD
```

Q&A

Thank you