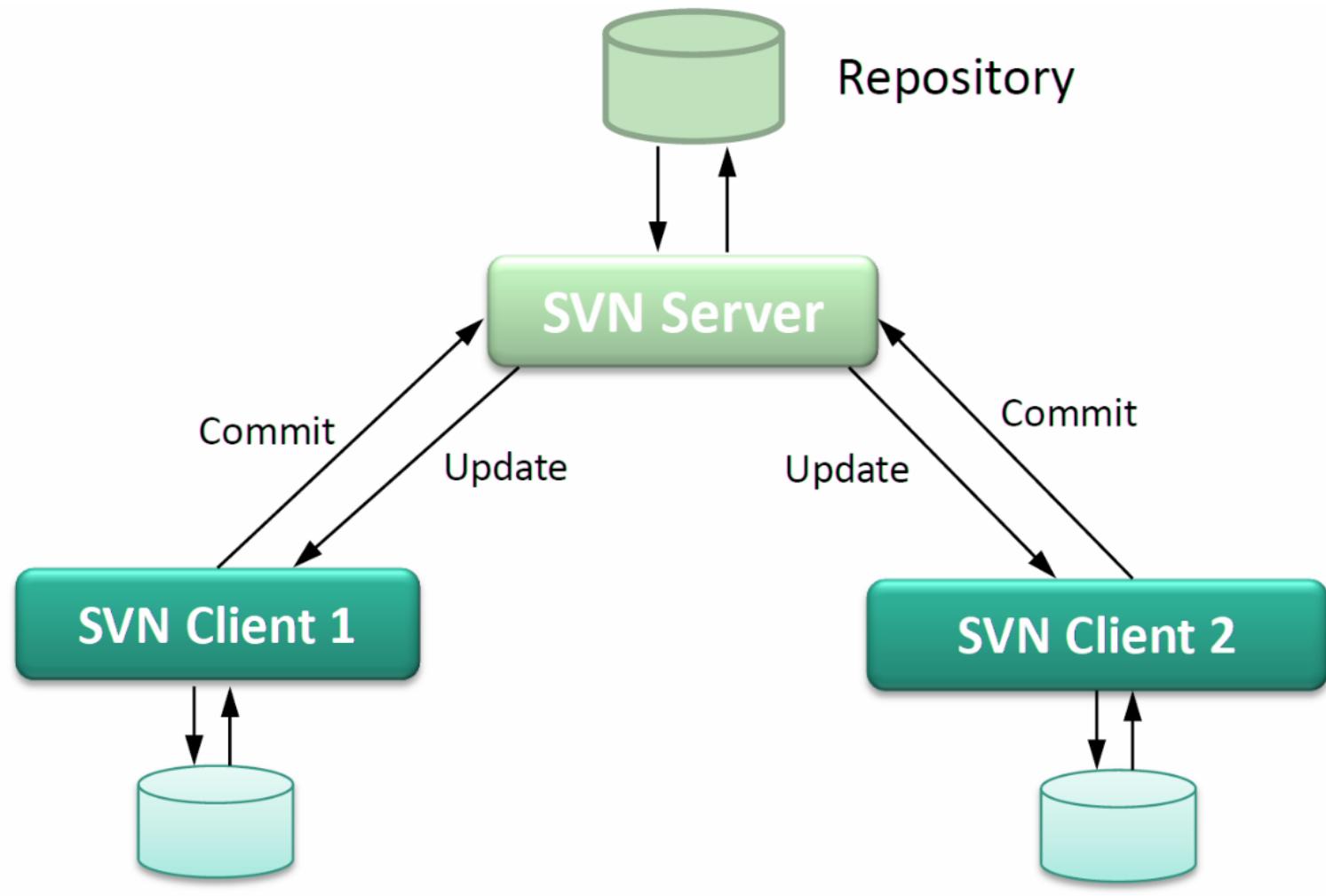

Git căn bản

cuong@techmaster.vn

Giới thiệu quản lý mã nguồn

Các cách quản lý mã nguồn

- **Sao lưu thư mục**, zip lại, đê tên theo ngày.
 - Ưu: ai cũng làm được, không cần tool, không cần học
 - Nhược: số lượng file tăng nhanh chóng, không cùng hợp tác code
- **Subversion**: quản lý mã nguồn mô hình client-server
 - Ưu: dễ hiểu, dễ dùng
 - Nhược: server chết, khỏi dùng!
- **Git**: phương pháp quản lý mã nguồn phổ biến nhất
 - Ưu: phổ biến nhất, nhiều dịch vụ free, nhiều lựa chọn client-server. Server chết, chập chờn, vẫn làm việc tốt
 - Nhược: tập lệnh phức tạp

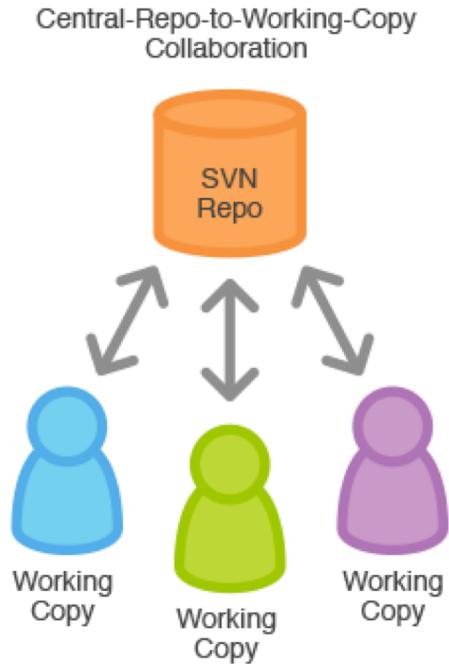


Local working copy 1

Local working copy 2

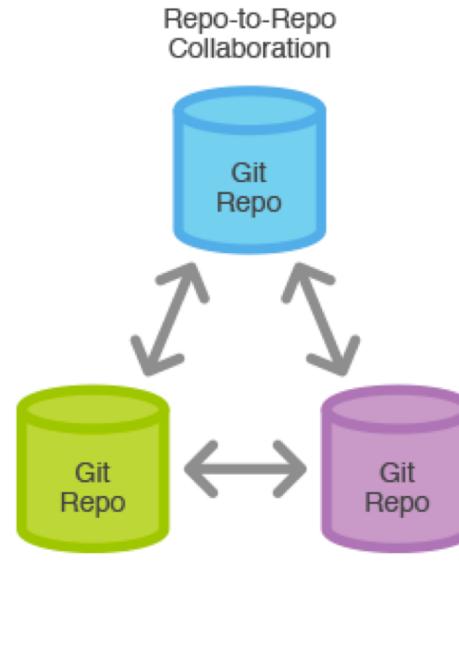
Subversion

- Tập trung hóa lên SVN server
- Ở client chỉ là bản sao

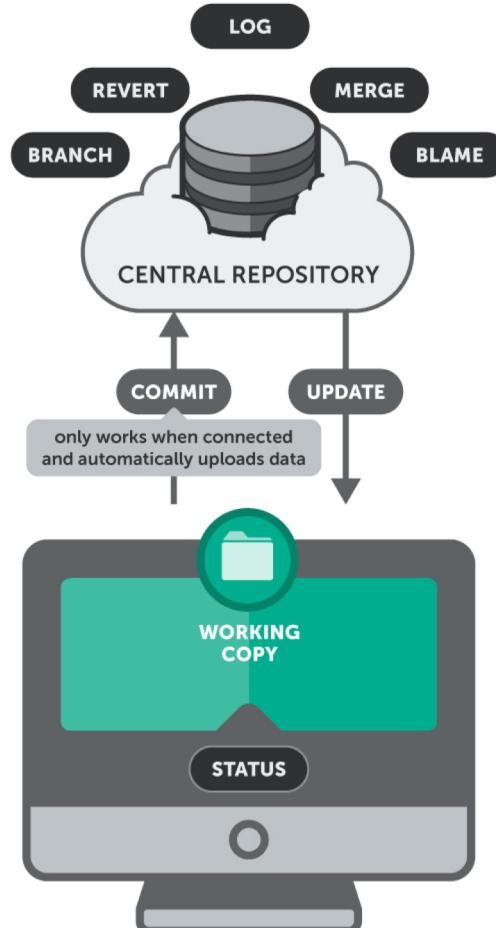


Git

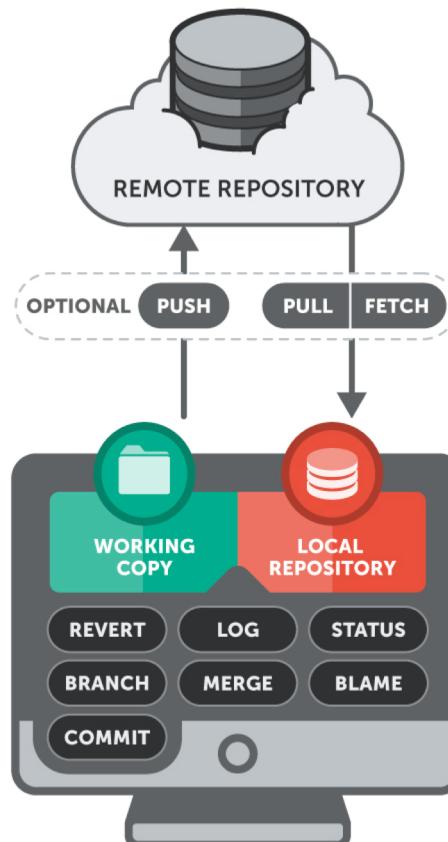
- Phân tán thành Git Repo:
 - Local Repository
 - Remote Repository



SUBVERSION

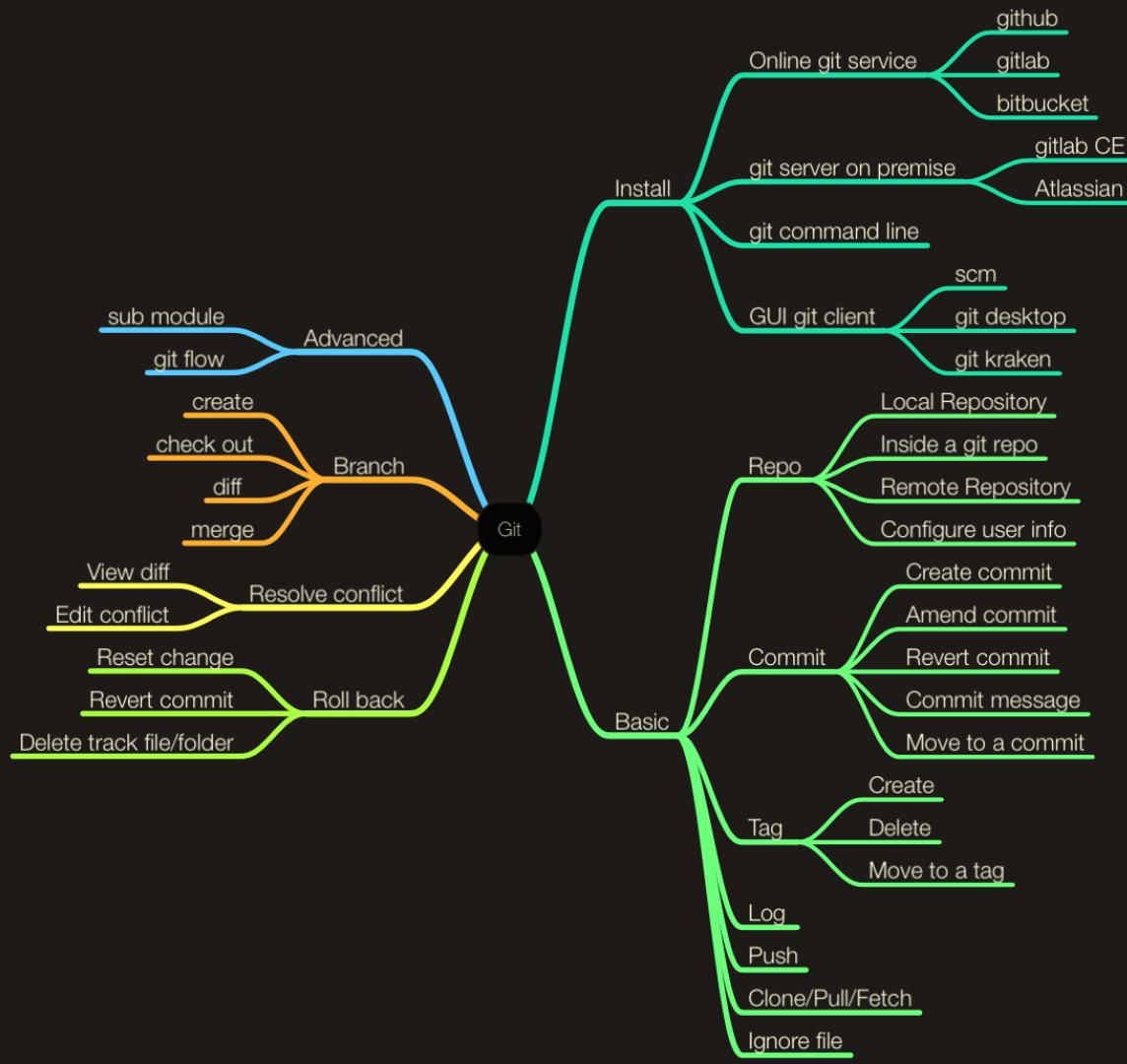


GIT



Tóm lại git là gì?

- Lưu vết lịch sử thay đổi mã nguồn ~ phiên bản
- Kết hợp quản lý mã nguồn tập trung sử dụng remote repo và phân tán ra các máy tram local repo
- Cấu hình remote repo mềm dẻo, cho phép có nhiều remote repo



Cài đặt git client

Cài đặt git client command line

- Windows
- Linux
- Mac

```
$ brew install git
```

Kiểm tra version hiện tại của git

```
$ which git
```

```
/usr/local/bin/git
```

```
$ git --version
```

```
git version 2.23.0
```

Git GUI client

- Dễ dùng hơn git command line vì có giao diện trực quan: xem các branch và commit
- Danh sách các git client
 - <https://git-scm.com/downloads/guis/>

swift
master

Fork* swift react*

Changes All Commits Starred master origin gsb-superclass Branches master master-next swift-5.0-branch test Remotes origin 4.1-dont-hardcode-n... 5.0-dont-hardcode-n... _fastCStringContents add-ninja-to-update-... anotherdayanotherco... asyncawait bananaphone builtin-int128 Character-test-patch cherr42 codable_test_comme... Compare-types-with-... core-team-resolution...

✓ master origin/master Merge pull request #20825 from DougGre... [Runtime] Make swift::swift_conformsToSwiftProtocol overridable. [Runtime] Add dummy "module name" parameter to swift::_conformsTo... Merge remote-tracking branch 'origin/master' into master-next

Merge pull request #20816 from harlanhaskins/the-missing-link [ParseableInterface] Test that module-link-name is serialized and used [ParseableInterface] Serialize module-link-name in binary module Merge remote-tracking branch 'origin/master' into master-next

Merge pull request #20717 from troughton/patch-2 Define mappings for CF types on LLP64 Merge pull request #20813 from drodriguez/fix-android-gold-link... Split link flags and link libraries lists to make gold happy.

Commit Changes File Tree

AUTHOR: Harlan Haskins harlan@apple.com
COMMITTER: GitHub noreply@github.com
SHA: 2889a1f906c481df1037b718bbb3b6289a8d6a27
PARENTS: e4efac5 717532e

Merge pull request #20816 from harlanhaskins/the-missing-link
[ParseableInterface] Ensure module-link-name is serialized and used

lib/Frontend/ParseableInterfaceSupport.cpp

git-fork

Expand All

Current Repository desktop

Current Branch esc-pr #3972 ✓

Fetch origin Last fetched 3 minutes ago

Changes History

Appease linter

iAmWillShepherd committed a day ago

Add event handler to dropdown com...

iAmWillShepherd and Markus Olsson...

Move escape behavior to correct co...

iAmWillShepherd and Markus Olsson...

Remove event handler from the bran...

iAmWillShepherd and Markus Olsson...

Merge branch 'master' into esc-pr

iAmWillShepherd committed a day ago

Merge pull request #4044 from des...

Neha Batra committed a day ago

Merge pull request #4070 from desk..

Brendan Forster committed 2 days ago

bump to beta3

Brendan Forster committed 2 days ago

Merge pull request #4057 from desk..

Brendan Forster committed 2 days ago

Merge pull request #4067 from desk..

Brendan Forster committed 2 days ago

Release to 1.1.0-beta2

Neha Batra committed 2 days ago

Add event handler to dropdown component

iAmWillShepherd and Markus Olsson committed c79e71c 1 changed file

Co-Authored-By: Markus Olsson <niik@users.noreply.github.com>

app/src/ui/t.../dropdown.tsx

```
@@ -145,6 +145,10 @@ export class ToolbarDropdown extends React.Component<
 145     this.state = { clientRect: null }
 146   }
 147 
+ private get isOpen() {
+   return this.props.dropdownState === 'open'
 148 
 149   private dropdownIcon(state: DropdownState): OcticonSymbol {
 150     // @TODO: Remake triangle octicon in a 12px version,
 151     // right now it's scaled badly on normal dpi monitor
 152   }
 153 
 154   @@ -249,6 +253,13 @@ export class ToolbarDropdown extends React.Component<
 249   }
 250   }
 251 
+ private onFoldoutKeyDown = (event: React.KeyboardEvent<HTMLElement>) => {
 252     if (!event.defaultPrevented && this.isOpen &&
 253       event.key === 'Escape') {
 254       event.preventDefault()
 255       this.props.onDropdownStateChanged('closed', 'keybo
 256     }
 257 
 258     + this.props.onDropdownStateChanged('closed', 'keybo
 259   }
```

git desktop

Commit Pull Push Branch Merge Shelve Show in Finder Terminal Settings

WORKSPACE

File status History Search

BRANCHES

BOOKMARKS

TAGS

REMOTES

SHELVED

SUBREPOSITORIES

All Branches Show Remote Branches Ancestor Order Jump to:

Graph	Commit	Author	Description	Date
b7358c7	Rahul Chhab...	master origin/master origin/HEAD	Removing ol...	Mar 3, 2016, 11:...
bdb8bef	Rahul Chhab...	Merged in update-google-verification (pull request #14)		Feb 18, 2016, 1:3...
dfe975d	Tyler Tadej...	origin/update-google-verification	Update google verificati...	Feb 11, 2016, 2:2...
3bc3290	Tyler Tadej...	Replace outdated Atlassian logo in footer with base-64 en...		Feb 11, 2016, 2:1...
dba47f9	Tyler Tadej...	Add gitignore		Feb 11, 2016, 1:3...
ff67b45	Mike Minns...	Updated Mac min-spec to 10.10		Feb 15, 2016, 11:...
72d32a8	Michael Min...	Merged in hero_images (pull request #13)		Feb 15, 2016, 10:...
246c4ff	Joel Unger...	origin/hero_images hero_images	Used Tinypng to c...	Feb 11, 2016, 3:3...
9d9438c	Joel Unger...	Replacing hero images with new version of SourceTree		Feb 9, 2016, 2:59...
ce75b63	Michael Min...	Merged in bug/date-https (pull request #12)		Feb 15, 2016, 10:...
85367bb	Patrick Tho...	origin/bug/date-https	fixed date and https errors	Jan 7, 2016, 12:2...
4f9b557	Joel Unger...	New Favicon		Feb 8, 2016, 3:55...
384e6d5	Rahul Chhab...	origin/search-console-access	search console google ver...	Feb 3, 2016, 2:09...
6fa47a9	Mike Minns...	updated to move supported version to OSX 10.9+		Dec 15, 2015, 2:0...
8dd87bb	Mike Minns...	remove extra , when a line is skipped due to empty server		Nov 23, 2015, 2:2...
faa195e	Mike Minns...	Skip records with empty server/user id as gas rejects them		Nov 23, 2015, 2:1...
0cdfe96	Mike Minns...	corrected paths after merge		Nov 23, 2015, 2:0...
051ab1b	Mike Minns...	corrected column counting		Nov 23, 2015, 1:5...
a723bc2	Mike Minns...	Merge branch 'au2gex'		Nov 23, 2015, 1:5...
65fd580	Mike Minns...	deal with invalid instanceids		Nov 23, 2015, 1:5...
500a892	Michael Min...	Merged in au2gex (pull request #11)		Nov 23, 2015, 1:0...

source tree

Visual Studio Code có git extension rất tốt

- Git Graph
- Git Lens

Buổi thực hành này tập trung làm việc
git command line và VSCode Git extension

Graph	Description	Date	Author	Commit
○	○ master fix d... 15 Oct 2019 00:04 techmaster cd472561			
●	Revert "Add line to ..."	14 Oct 2019 23:59	techmaster	6ebe7711
●	Break demo.py	14 Oct 2019 23:59	techmaster	63752344
●	b4 Fix demo.py	14 Oct 2019 23:52	techmaster	0257db84
●	branch3 Add li...	14 Oct 2019 22:09	techmaster	30be7424
●	Add line to ReadMe....	14 Oct 2019 21:53	techmaster	0b474caf
●	3rd commit	14 Oct 2019 18:41	techmaster	aafe3008
●	2nd commit	14 Oct 2019 18:32	techmaster	08a30229
●	branch2 Add R...	14 Oct 2019 11:56	techmaster	5acc6334

Tạo local git repo



Tình huống

- Bạn bắt đầu một dự án mới tinh cần quản lý mã nguồn dự án này.
- Mã nguồn ban đầu chỉ có file ReadMe.md (định dạng Markdown)
- Sau đó thêm file demo.py (Python script)

Git repository là gì?

- Nơi quản lý lịch sử sửa đổi file mã nguồn ~ phiên bản
- Local Repo
- Remote Repo

Tạo một local git repo

```
$ mkdir myweb
```

Tạo một thư mục

```
$ cd myweb
```

Chuyển vào thư mục đó

```
$ git init
```

Khởi tạo git repo

```
$ la
```

Liệt kê nội dung thư mục kể cả file, folder ẩn

```
.git
```

Thấy ngay một thư mục ẩn .git

Thêm một file

\$ nano ReadMe.md

Tạo một file ReadMe.md bằng trình editor nano

\$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

ReadMe.md



Xem trạng thái của git repo hiện tại

File mới tạo chưa được giám sát (untracked)

Staging và commit file vào repo

\$ git add .

Staging để giám sát thay đổi file này.
Dấu . lấy tất nội dung trong thư mục hiện tại

\$ git status

Kiểm tra trạng thái git repo

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)
new file: ReadMe.md

\$ git commit -m "Tạo ReadMe.md"

Chốt thay đổi vào git repo

[master (root-commit) 31f9ba8] Tạo file ReadMe.md

1 file changed, 1 insertion(+)

create mode 100644 ReadMe.md

Xem lịch sử thay đổi



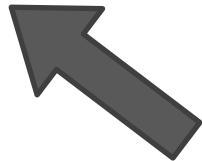
```
$ git log
```

```
commit 31f9ba8ca7572347121dacbad102ad9392edda45 (HEAD -> master)
```

```
Author: techmaster <cuong@techmaster.vn>
```

```
Date: Mon Oct 14 11:56:11 2019 +0700
```

Tạo file ReadMe.md
(END)



commit message

Chú ý: gõ vào terminal :q để thoát

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

```
31f9ba8 (HEAD -> master) Tạo file ReadMe.md  
(END)
```

Sửa lỗi → commit

Sửa lỗi → hủy sửa lỗi



Tình huống

User sửa file ReadMe.md, thêm vào vài dòng text đồng thời tạo mới file demo.py

Thêm file mới và sửa file đã commit

```
$ git status
```

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: demo.py File thêm mới

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: ReadMe.md File sửa đổi

Có 4 khả năng user muôn

1. Sửa nhầm file ReadMe.md, nên phải hủy thay đổi nhầm
2. Muốn commit thay đổi file ReadMe.md và thêm cả file mới tạo demo.py
3. Chỉ commit thay đổi file ReadMe.md, không commit file demo.py
4. Hủy tất cả thay đổi với ReadMe.md và demo.py, đưa thư mục làm việc về trạng thái lần commit trước đó

TH1: Hủy thay đổi lên file ReadMe.md

```
$ cat ReadMe.md
```

Hello World

Thêm thay đổi

```
$ git restore ReadMe.md
```

```
$ cat ReadMe.md
```

Hello World

Xem nội dung file ReadMe.md

Hủy thay đổi, đưa nội dung về lần commit gần nhất

Xem nội dung file ReadMe.md để thấy dòng chữ sửa đổi đã mất

TH2: staging cả ReadMe.md và demo.py

```
$ git add demo.py ReadMe.md
```

Cách này ghi rõ từng file

```
$ git status
```

On branch master

Changes to be committed:

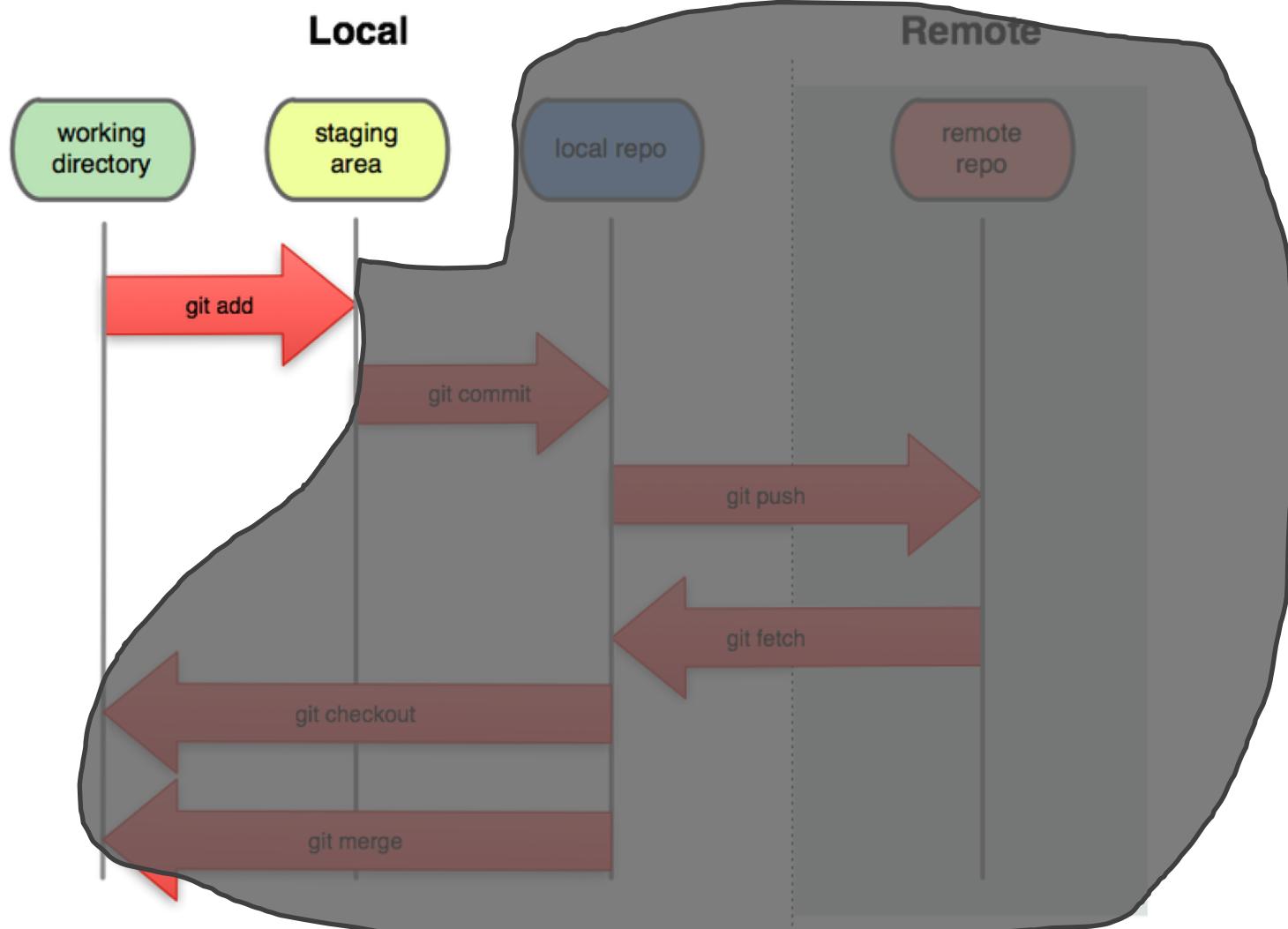
(use "git restore --staged <file>..." to unstage)

modified: ReadMe.md

new file: demo.py

```
$ git add .
```

Cách này stage tất cả thêm, sửa, xóa



Giữa working directory và local repo là staging area

- Working directory là thư mục mã nguồn dev đã sửa đổi
- Staging area là vùng đệm. Git chỉ commit những file/folder ở staging area

```
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   ReadMe.md
    new file:   demo.py

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   ReadMe.md
```



File ReadMe.md đã
staging rồi lại bị sửa thêm
lần nữa

Hỏi: muốn hủy nội dung đã lên staging thì làm thế nào?

```
$ git restore --staged ReadMe.md
```

```
$ git status
```

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)
new file: demo.py

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working
directory)

modified: ReadMe.md

} Staging area

} Working directory

TH3: Chỉ commit thay đổi ReadMe.md bỏ qua demo.py

```
$ git status
```

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: ReadMe.md

new file: demo.py

```
$ git restore --staged demo.py
```

Loại demo.py ra khỏi staging

```
$ git status
```

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: ReadMe.md

Untracked files:

(use "git add <file>..." to include in what will be committed)

demo.py

TH4: Hủy tất cả thay đổi, kể cả staging area đưa working directory về nội dung commit mới nhất

\$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: ReadMe.md

new file: demo.py

\$ git reset --hard

HEAD is now at 31f9ba8 Tạo file ReadMe.md

\$ git status

On branch master

nothing to commit, working tree clean

**.gitignore loại file, thư mục
không cần quản lý phiên bản**



Tình huống

Hãy tạo thêm một thư mục temp cùng một số file bên trong và một file junk.csv ở ngoài

```
$ mkdir temp  
$ touch temp/1.csv  
$ touch temp/2.csv  
$ touch junk.csv  
$ tree .
```

```
.  
├── ReadMe.md  
└── junk.csv  
└── temp  
    ├── 1.csv  
    └── 2.csv
```

1 directory, 4 files

Không muốn quản lý thư mục temp và file tạm junk.csv thì làm sao?

Hãy tạo một file .gitignore có nội dung như sau

```
temp/  
junk.csv
```

```
$ git status
```

On branch master
Untracked files:

(use "git add <file>..." to include in what will be committed)

.gitignore

Giờ thì thư mục temp và file junk.csv đã bỏ qua trong các lệnh git add

.gitignore có đặc điểm gì?

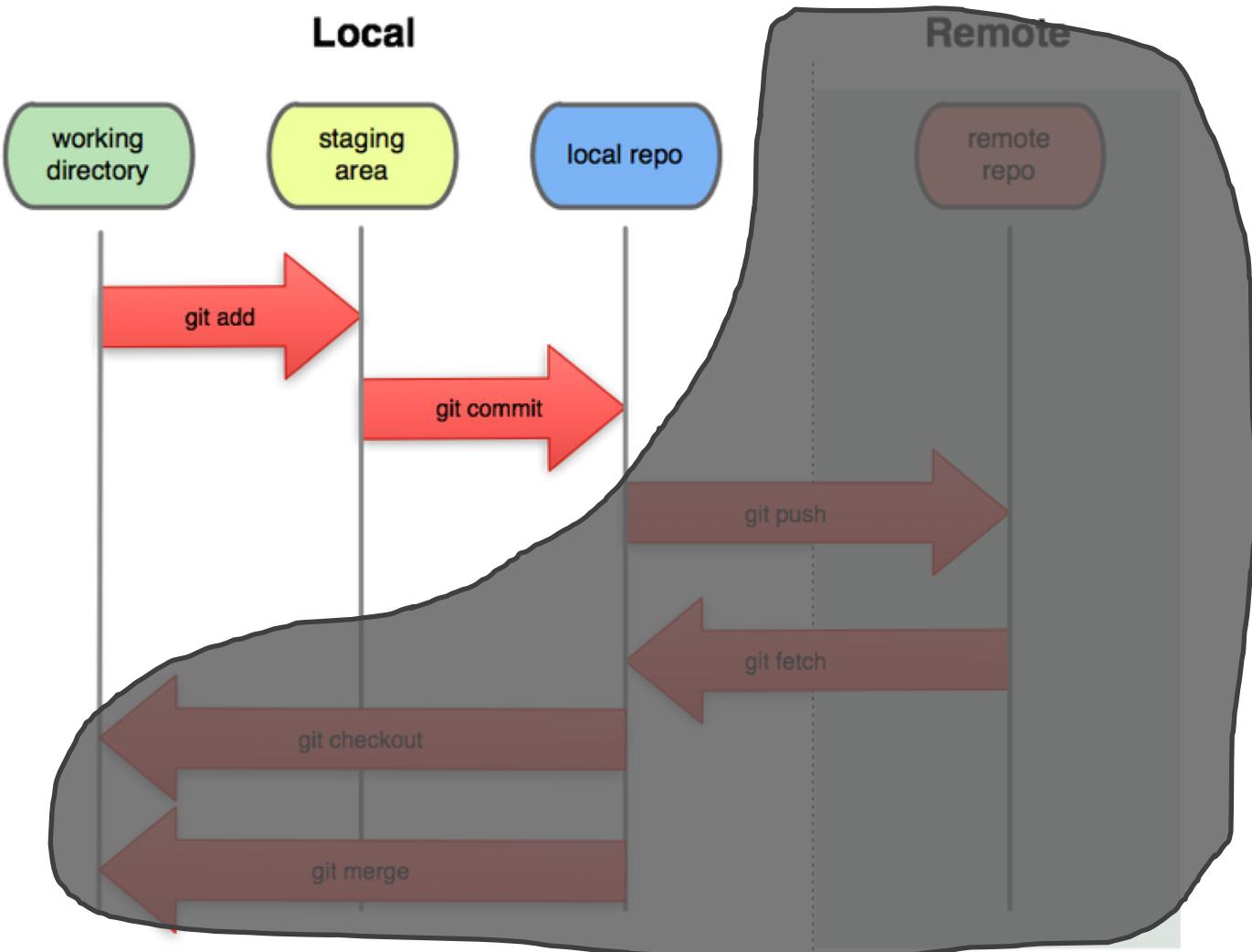
- Hỗ trợ wild card *.file_extension
- Có thể tạo .gitignore trong thư mục con để loại bỏ chi tiết hơn
- Dùng để bỏ qua những file quá lớn, những file tạm không có ý nghĩa trong việc quản lý phiên bản

git commit



Tình huống

- Ở phần thực hành trước chúng ta mới làm việc ở working directory và staging area, giờ hãy tìm hiểu tiếp commit vào local repo
- Commit là một tập các thay đổi được người chốt hạ thành một phiên bản
- Staging có thể quay lui, thì commit cũng có thể quay lui



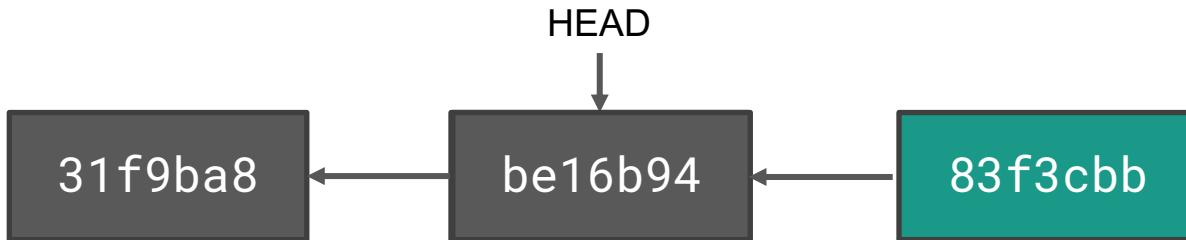
Commit lần 2 vào local repo

```
$ git add .
```

```
$ git commit -m "2nd commit"
```

```
[master be16b94] 2nd commit
 2 files changed, 3 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 demo.py
```

```
$ git log --oneline  
be16b94 (HEAD -> master) 2nd commit  
31f9ba8 Tạo file ReadMe.md
```



Commit best practice

- Mỗi lần commit tập trung vào một nhiệm vụ: fix 1 bug, refactor một số đoạn code, thêm một chức năng gọi tên cụ thể
- Không nên tạo big commit gồm quá nhiều thay đổi
- Không đặt tên commit message vô nghĩa
 - “fix something” → “fix issue #2301”
 - “Add some file” → add ReadMe.md, toc.md
 - “Make big change” → “refactor Person.go”

git commit --amend



Tình huống

Tôi vừa commit xong vào local repo, chưa push code lên remote repo.
Tôi muốn chỉnh lại commit message cho rõ ràng hơn.

```
$ git commit -m "bad message"  
[master 4d6410e] bad message  
 1 file changed, 1 insertion(+)  
$ git commit --amend  
[master 3d0e28f] Toi da sua file index.html  
Date: Fri Oct 18 18:05:31 2019 +0700  
 1 file changed, 1 insertion(+)
```

```
$ git commit --amend
```

```
[master 0138088] Them index.html
Date: Sat Oct 19 15:02:16 2019 +0700
1 file changed, 8 insertions(+)
create mode 100644 index.html
```

```
$ git log -oneline
```

```
// Khi push commit đã amend sẽ bị báo lỗi
```

```
$ git push -u origin master
```

```
To http://0.0.0.0:8081/vinid/myblog
```

```
! [rejected]          master -> master (non-fast-forward)
```

```
error: failed to push some refs to 'http://0.0.0.0:8081/vinid/myblog'
```

```
hint: Updates were rejected because the tip of your current branch is behind
```

```
hint: its remote counterpart. Integrate the remote changes (e.g.
```

```
hint: 'git pull ...') before pushing again.
```

```
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

```
$ git pull
```

```
// Chỗ này sẽ phải resolve conflict và merge
```

```
$ git push -u origin master
```

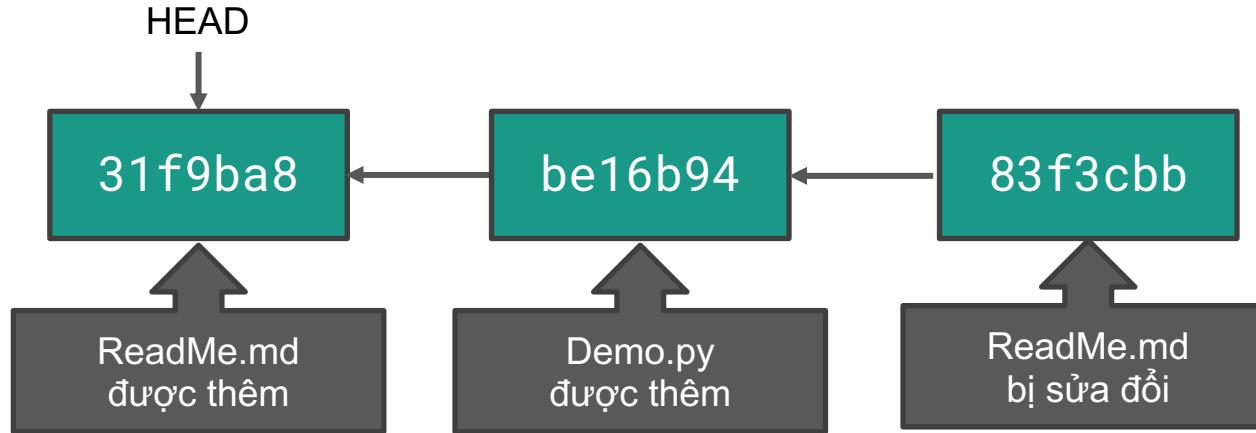
git checkout



Dùng git checkout để làm gì?

- Lật lại mã nguồn trong quá khứ, tìm nguyên nhân gây lỗi
- Hoặc code hiện tại sau khi sửa phát sinh quá nhiều lỗi, cần quay về code ổn định trước đây

Ôn lại quá trình commit



Mỗi commit được gắn với một chuỗi ký tự unique – gọi là hash string

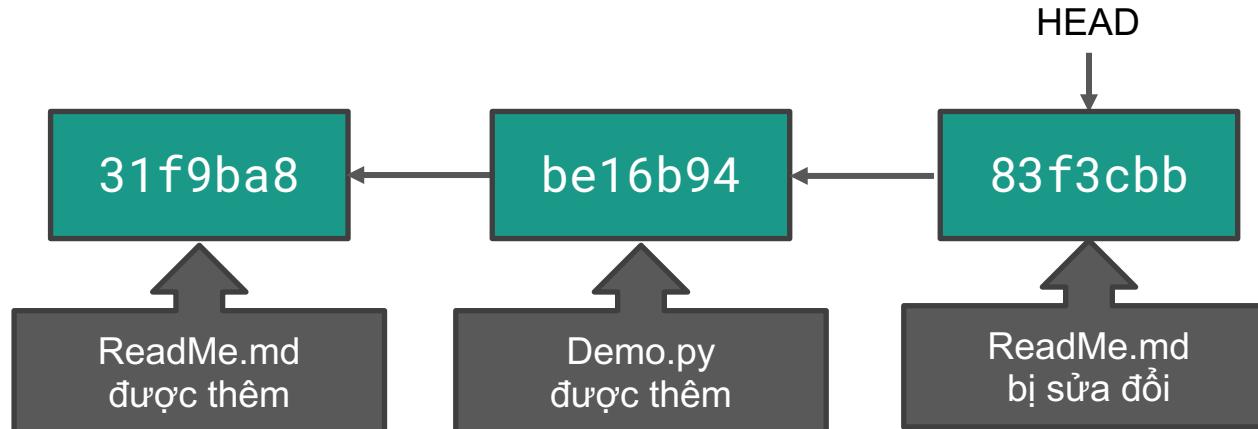
Thực ra chuỗi này dài nhưng khi gõ git log -oneline ta chỉ thấy chuỗi rút gọn

Để di chuyển, tìm lại nội dung mã nguồn tại một thời điểm trong quá khứ, chúng ta cần đến chuỗi rút gọn này.

git checkout chuyển đến commit cụ thể

Nội dung working directory mới nhất sẽ bị thay thế bằng nội dung của working directory tại thời điểm commit git trước đây. Nhưng file/folder bị lờ đi trong .gitignore không bị thay đổi

```
$ git checkout 31f9ba8
```



Quay về commit mới nhất

Sau xem xét, tìm hiểu commit cũ có 2 khả năng:

1. User không thay đổi gì và muốn đưa working folder về lần commit gần đây nhất

```
$ git checkout -
```

2. User thay đổi nội dung file → có 2 khả năng:

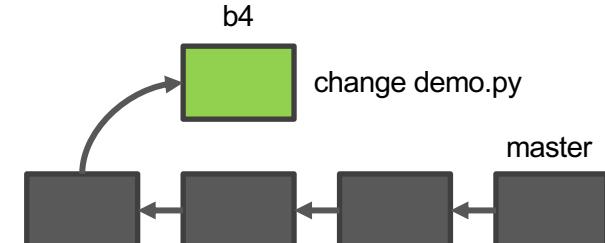
- a. Hủy thay đổi này: `$ git restore .`

- b. Commit thay đổi này, rõ ràng không thể vào nhánh hiện tại mà phải rẽ nhánh

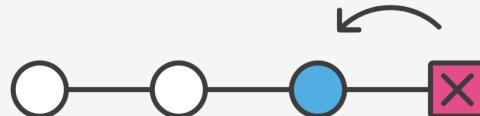
```
$ git branch -c b4
```

```
$ git add .
```

```
$ git commit -m "change demo.py"
```



Git reset



Tình huống

Sau khi commit vài lần những thay đổi hết sức ngớ ngẩn, tôi muốn viết lại lịch sử, loại bỏ hẳn các commit ngớ ngẩn vĩnh viễn

Nhận xét:

- Nếu chưa pushed lên remote repo, đây là cách gọn gang sạch sẽ
- Nếu đã pushed lên remote repo, cách này gây xung đột

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

```
c8a4b05 (HEAD -> b4) Fix demo.py
```

```
00750f8 Shit demo.py
```

```
6e1c7fe Change demo.py
```

```
08a3022 2nd commit
```

```
5acc633 (branch2) Add ReadMe.md
```

```
$ git reset --hard 08a3022
```

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

```
08a3022 (HEAD -> b4) 2nd commit
```

```
5acc633 (branch2) Add ReadMe.md
```



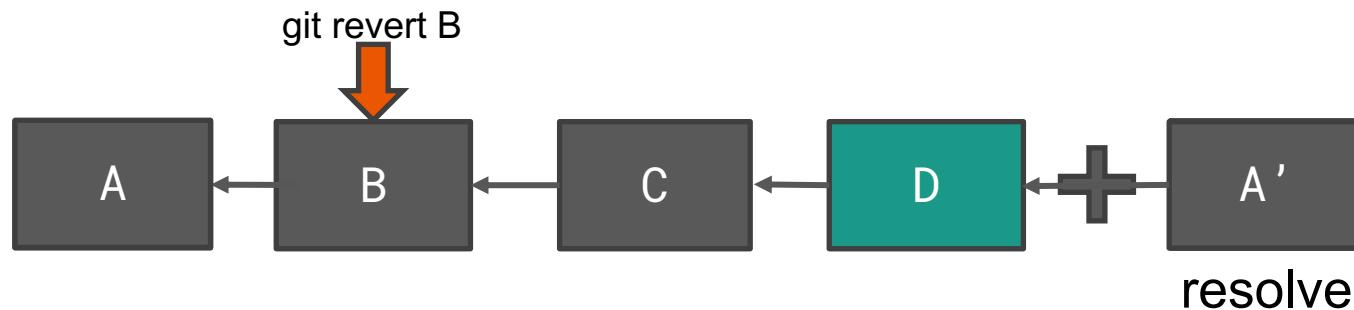
git reset --hard 08a3022

Reset vs revert

Tình huống

Tôi tạo ra commit $A \leftarrow B \leftarrow C \leftarrow D$. Tôi muốn lấy lại code của A nhưng không muốn xóa các commit B, C, D khỏi lịch sử.
Nếu dùng git reset thì phũ quá.

Vậy hãy dùng `$ git revert B`



```
$ git revert commit_hash
```

```
$ git status
```

Resolve conflict

```
$ git add .
```

```
$ git revert --continue
```

```
1 print("Hello World")
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
2 <<<<< HEAD (Current Change)
3 # quay lại máng lợn
4 =====
5 >>>>> parent of aafe300... 3rd commit (Incoming Change)
6
```

Ảnh chụp VSCode tại bước resolve conflict:

- Incoming change là nội dung của commit cũ A
- Current change là nội dung của commit gần nhất

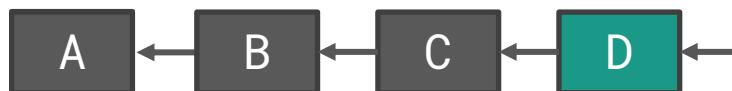
reset

- Xóa các commit ra khỏi lịch sử của local repo
- Có thể gây lộn xộn, mất đồng bộ với remote repo



revert

- Không xóa commit cũ, mà chỉ tạo commit mới có nội dung lấy từ commit cũ
- Cần phải resolve conflict
- Không gây xung đột với remote repo



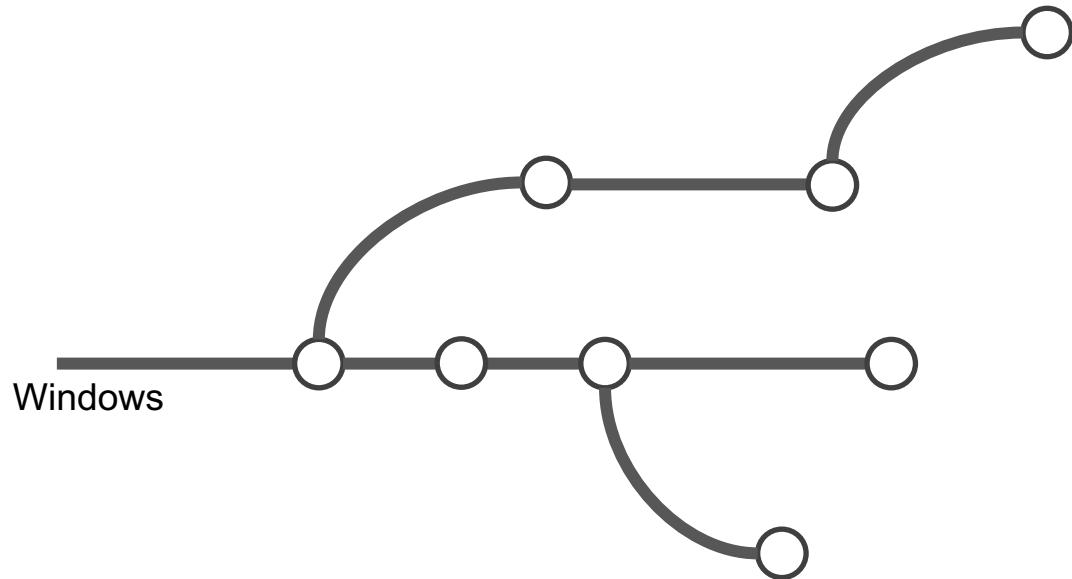
Xem thêm

<https://dev.to/neshaz/when-to-use-git-reset-git-revert--git-checkout-18je>

Git branch

Tình huống

Khi nào sử dụng nhánh (branch)?



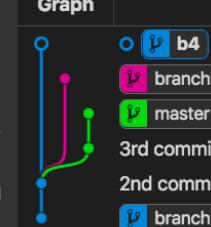
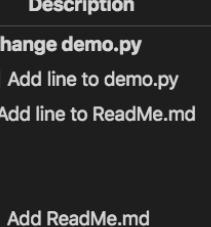
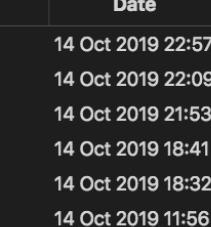
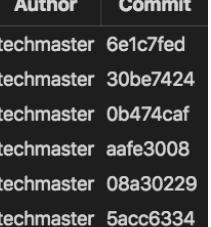
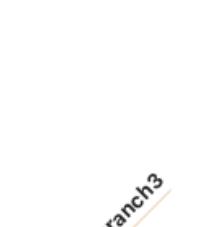
```
* b4  
branch2  
branch3  
master  
(END)
```

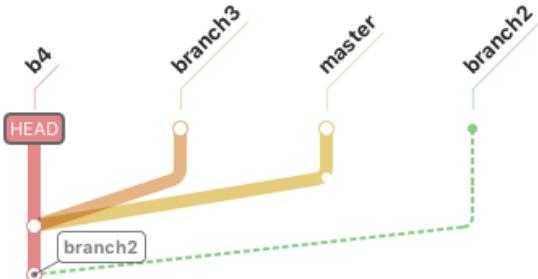
\$ git branch

Git Graph — gitdemo

demo.py (Uncommitted)

Branches: Show All Show Remote Branches

Graph	Description	Date	Author	Commit
	b4 Change demo.py	14 Oct 2019 22:57	techmaster	6e1c7fed
	branch3 Add line to demo.py	14 Oct 2019 22:09	techmaster	30be7424
	master Add line to ReadMe.md	14 Oct 2019 21:53	techmaster	0b474caf
	3rd commit	14 Oct 2019 18:41	techmaster	aafe3008
	2nd commit	14 Oct 2019 18:32	techmaster	08a30229
	branch2 Add ReadMe.md	14 Oct 2019 11:56	techmaster	5acc6334



Làm việc với Remote Repo

Tình huống

A: nén code lại gửi tôi qua Zalo nhé

B: Vừa gửi qua gmail rồi mà

C: Ờ thế sao không đẩy lên Google Drive hay DropBox?

Hôm sau:

A: Tôi vừa fix lỗi, gửi lại code qua Zalo này. Tải về đi !



Remote git repository

To be able to collaborate on any Git project, you need to know how to manage your remote repositories. Remote repositories are versions of your project that are hosted on the Internet or network somewhere. You can have several of them, each of which generally is either read-only or read/write for you. Collaborating with others involves managing these remote repositories and pushing and pulling data to and from them when you need to share work. Managing remote repositories includes knowing how to add remote repositories, remove remotes that are no longer valid, manage various remote branches and define them as being tracked or not, and more. In this section, we'll cover some of these remote-management skills.

Làm việc với remote repository

Liệt kê remote repo

```
$ git remote -v
```

Tên remote repo

Thêm remote repo



```
$ git remote add origin remote_repo_URL
```

Đẩy code từ local repo lên remote repo

```
$ git push -u origin master
```

Xóa remote repo

```
$ git remote remove origin
```

Fetch vs Pull code từ remote repo

Cài đặt GitLab bằng Docker

Tình huống

Đội lập trình cần dựng một GitLab server trong mạng LAN, được bảo mật để tránh những truy cập từ bên ngoài.

Tại sao team chọn GitLab?

- Giao diện thân thiện
- Dễ cài đặt thông qua Docker
- Có 2 lựa chọn Community Edition và Enterprise Edition

Cài đặt Gitlab qua Docker

Cài đặt Docker trên Linux hay Windows

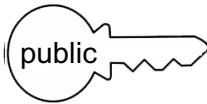
```
$ docker run --detach \
--publish 4430:443 --publish 8081:80 --publish 2200:22 \
--name gitlab \
--restart always \
--volume ~/gitlab/config:/etc/gitlab \
--volume ~/gitlab/logs:/var/log/gitlab \
--volume ~/gitlab/data:/var/opt/gitlab \
gitlab/gitlab-ee:latest
```

Cài xong truy cập vào <http://0.0.0.0:8081>

Gitlab hay GitHub sử dụng SSH public, private key để không phải sử dụng user-name, password khi push code lên remote repo



- 1 Generate public & private SSH key



- 3 Encrypt message with private key



- 2 Send public SSH key to Git Server

- 4 Send to server

- 5 Decrypt message by public SSH key



Fingerprint: d1:bb:f7:de:fb:4d:31:fe:96:4c:43:5d:c7:1f:e6:39

ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAIhFuTMHeHnhBq4FudF/Ay5687SBEvkmh1vlHJz5nJs



Remove

Copy public SSH key được sinh từ client vào mục SSH key
trong User Settings > SSH Keys

clone một remote repo trên Internet

```
$ git clone xxxxx
```

- Tập trung vào
 - thư mục .git
 - .gitignore
 - ReadMe.md

Git repo

- Lưu vết lịch sử commit, nội dung commit

Working Copy

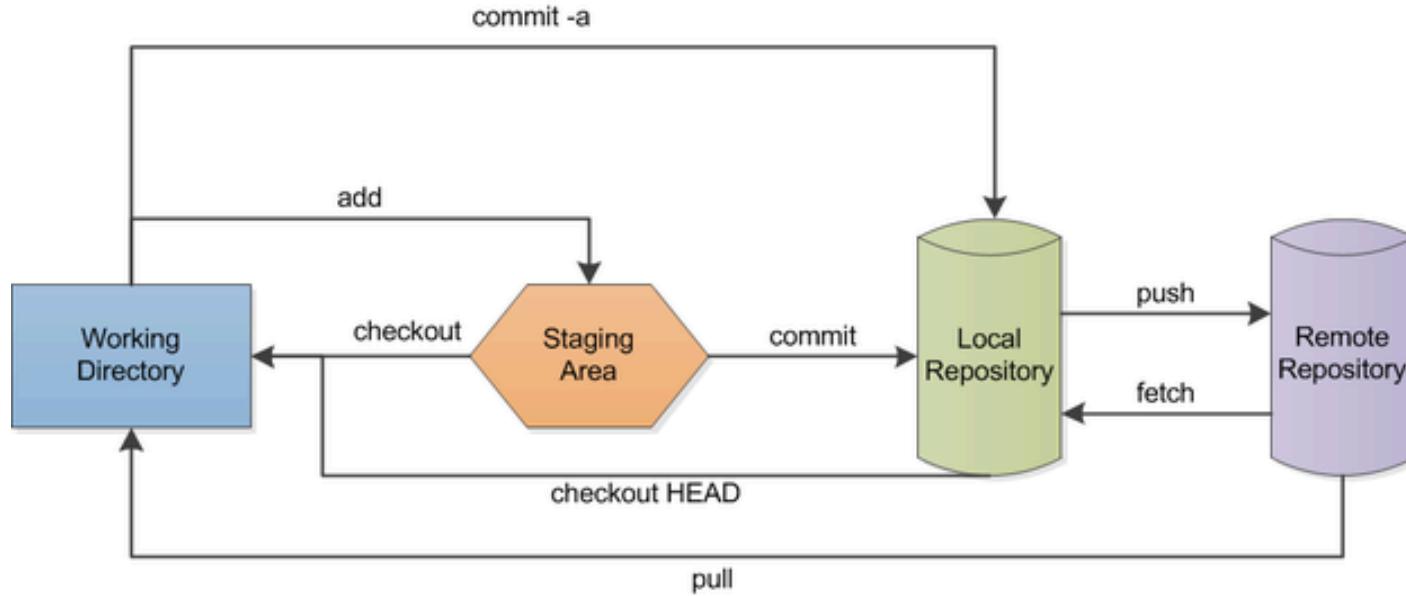
- Nội dung mã nguồn ở tại một phiên bản (commit) nào đó. Có thể là
 - Commit trong quá khứ
 - Hoặc phiên bản sửa đổi của commit mới nhất

Local git repo khác gì với remote git repo

What makes git so awesome is it's a distributed version control system. **Your local repository has exactly the same features and functionality as any other git repository.** So a git repo on a server is the same as a git repo on github (granted github adds additional features, but at its core you're dealing with git repos) which is the same as your coworker's local repo.

So why is that awesome? Because there is no central repo you have to have access to to do your work. **You can commit, branch and party on your own repo on your local machine even without internet access.** Then, when you have a connection again, you can push your changes to any other git repo you have access to. So while most people treat a particular repo as the central repo, that's a process choice, not a git requirement.

Quay lui thay đổi



commit -a: Directly commit modified and deleted files into the local repository (*no new files!*)
add: Add a file to the staging area.

checkout: Get a file from the staging area.

checkout HEAD: Get a file from the local repository

commit: Commit files from the staging area to the local repository

push: Send files to the remote repository

fetch: Get files from the remote repository

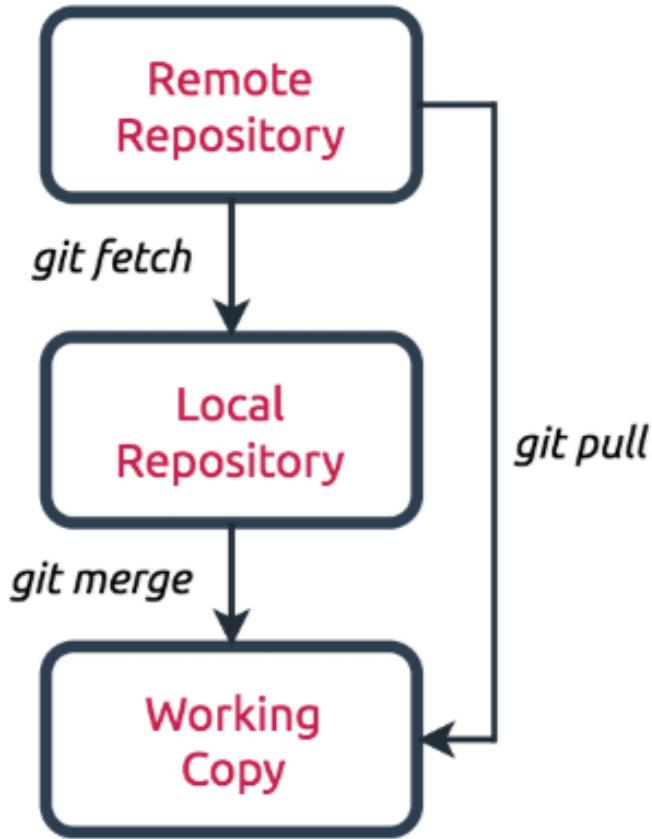
pull: Get files from the remote repository and put a copy in the working directory

Hợp tác code qua Git

Tình huống

- 3 user: Tom, Jerry, Mick cùng nhau lập trình một web site
 - Tom chuyên HTML
 - Jerry chuyên CSS nhưng khi cần vẫn sửa HTML
 - Mick chuyên viết document và làm tester
- Dùng Gitlab để lưu remote repo web site do Tom, Jerry, Mick lập trình

git pull vs git fetch



Sửa commit message



Có nhiều trường hợp

Tham khảo <https://gist.github.com/nepsilon/156387acf9e1e72d48fa35c4fabef0b4>

1. Not pushed + latest commit
2. Pushed + latest commit
3. Not pushed + old commit
4. Pushed + old commit