[Git] Hiểu thêm về Commit và Staging Area

Cập nhật lúc 17/09/2016 | No Comments | Công cụ, Web Development

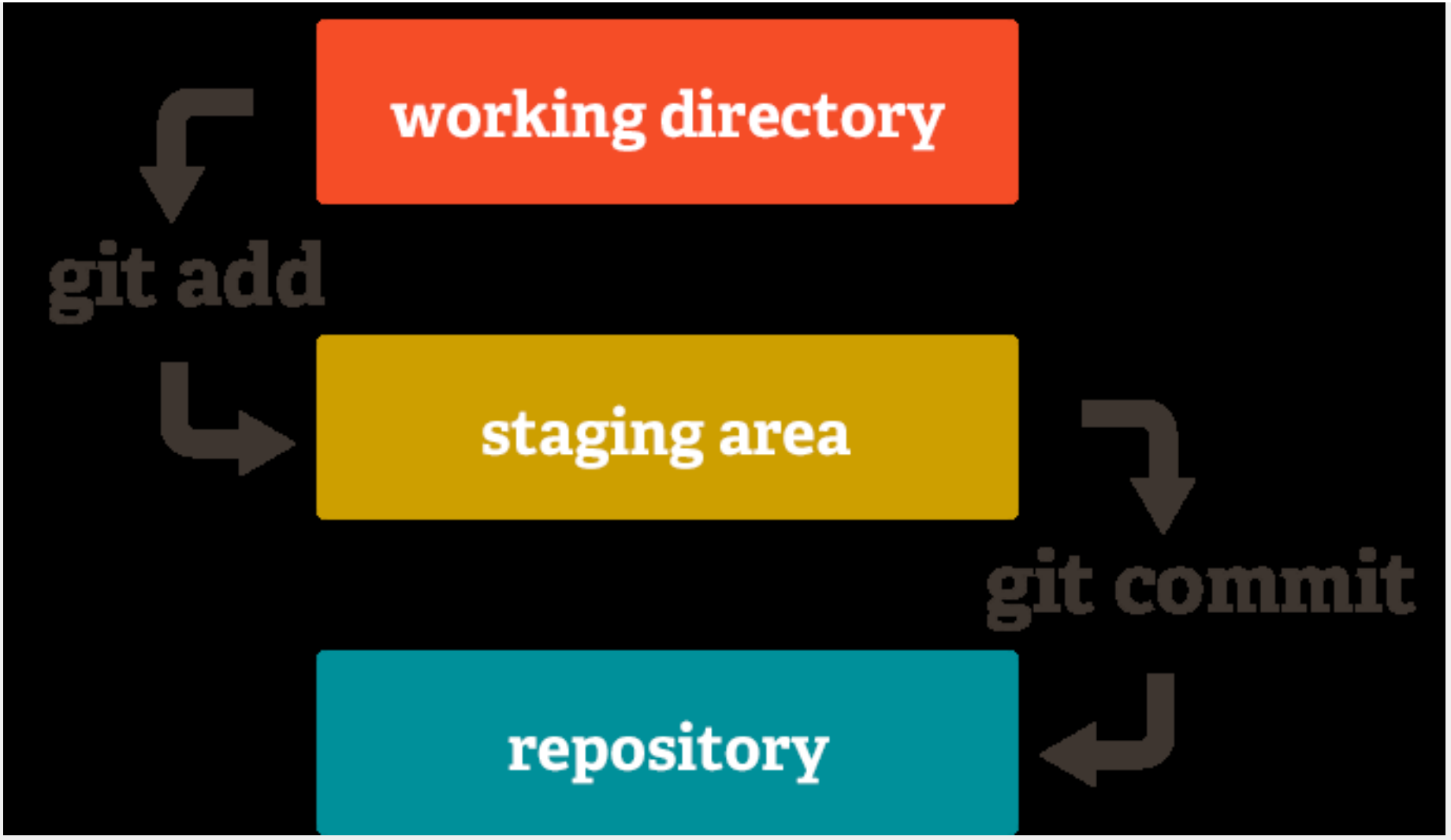
Bài này thuộc phần 5 của 11 phần trong serie **Git cơ bản**

Trong bài tạo repository cho Git mình có nhắc qua về cụm từ **Staging Area** và một tính năng là **commit** (ủy thác), vậy hai cái này là gì thì mình sẽ giải thích kỹ hơn trong bài này để bạn biết cách sử dụng cho đúng.

Staging Area là gì?

Staging Area nghĩa là một khu vực mà nó sẽ được chuẩn bị cho quá trình commit. Trước hết, bạn cần phải hiểu rằng trong các hệ thống quản lý phiên bản (Version Control System) thì các dữ liệu sẽ được lưu trữ ở hai nơi, một là thư mục bạn đang làm việc trên máy tính (working tree, mình không nhắc lại nữa đâu) và một là kho chứa mã nguồn (repository) sau khi bạn đã thực hiện thay đổi (ví dụ như kho chứa trên Github).

Nhưng với Git thì nó có thêm một lựa chọn nữa đó là có thêm một khu vực trung gian gọi là **Staging Area** và đây chính là một lợi thế lớn của Git. Staging Area nghĩa là khu vực sẽ lưu trữ những thay đổi của bạn trên tập tin để nó có thể được commit, vì muốn commit tập tin nào thì tập tin đó phải nằm trong Staging Area. Một tập tin khi nằm trong Staging Area sẽ có trạng thái là **Stagged** (xem thêm ở dưới).



Mô hình giải thích cách hoạt động của Staging Area.

Và để đưa một tập tin vào Staging Area thì bạn sẽ cần phải sử dụng lệnh `git add tên_file` mà mình đã có ví dụ ở phần trước.

Commit là gì và nó hoạt động ra sao?

Hiểu đơn giản hơn, commit nghĩa là một hành động để Git lưu lại một bản chụp (snapshot) của các sự thay đổi trong thư mục làm việc, và các tập tin và thư mục được thay đổi đã phải nằm trong Staging Area. Mỗi lần commit nó sẽ được lưu lại lịch sử chỉnh sửa của mã nguồn kèm theo tên và địa chỉ email của người commit. Ngoài ra trong Git bạn cũng có thể khôi phục lại tập tin trong lịch sử commit của nó để chia cho một phân nhánh (branch) khác, đây là mấu chốt của việc bạn sẽ dễ dàng khôi phục lại các thay đổi trước đó mà mình có giới thiệu qua ở phần giới thiệu serie này.

Và tất nhiên, lệnh commit trong Git sẽ là `git commit -m "Lời nhắn"`.

Và nếu bạn **muốn đưa tập tin lên repository thì bạn phải commit nó trước** rồi sau đó lệnh `git push origin master` sẽ có nhiệm vụ đưa toàn bộ các tập tin đã được commit lên repository.

Điều kiện gì để commit một tập tin?

Nếu bạn muốn commit một tập tin đó, bạn sẽ cần phải đưa tập tin đó vào trạng thái tracked bằng lệnh `git add tên_file`. Trong git có hai loại trạng thái chính đó là Tracked và Untracked, cụ thể:

- **Tracked** – Là tập tin đã được đánh dấu theo dõi trong Git để bạn làm việc với nó. Và trạng thái Tracked nó sẽ có thêm các trạng thái phụ khác là Unmodified (chưa chỉnh sửa gì), Modified (đã chỉnh sửa) và Staged (đã sẵn sàng để commit).

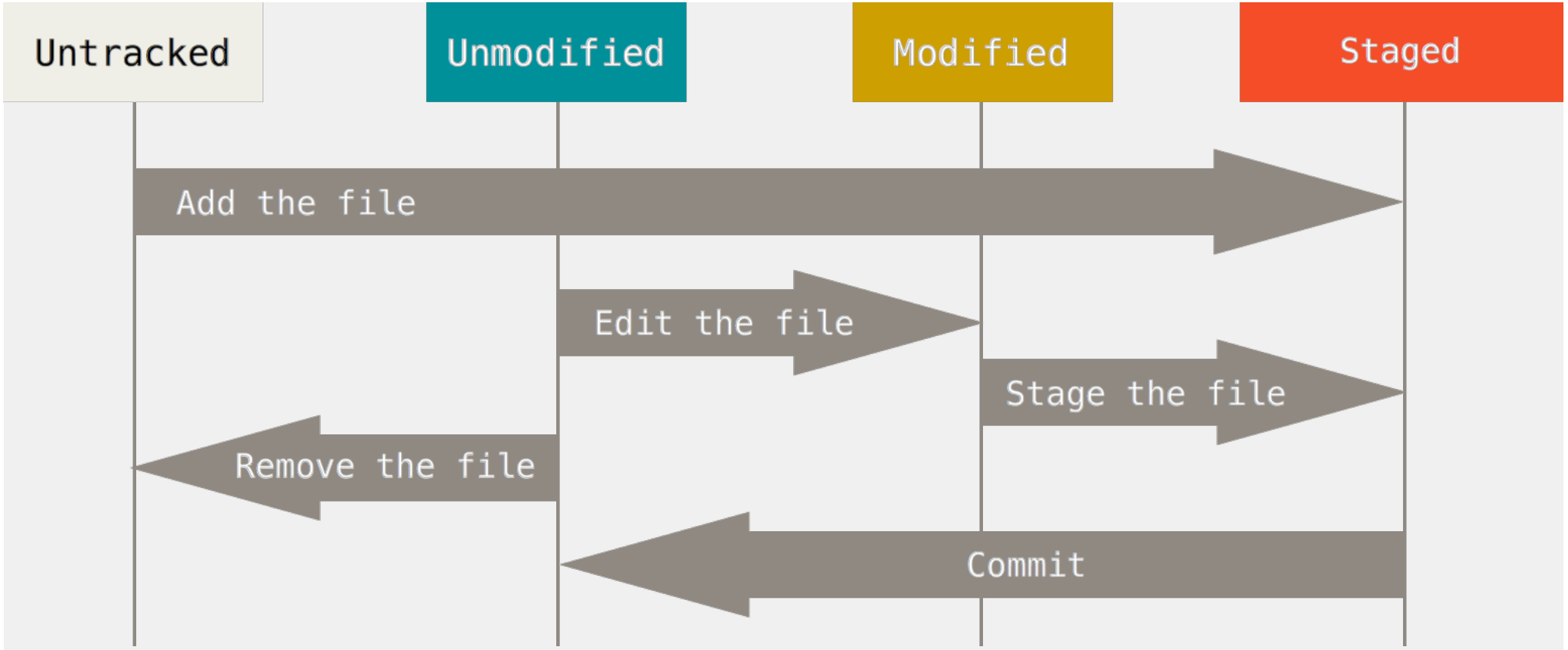
- **Untracked** – Là tập tin còn lại mà bạn sẽ không muốn làm việc với nó trong Git.

Nhưng bạn phải nên biết rằng nếu tập tin đó đã được Tracked nhưng đang rơi vào trạng thái (Modified) thì nó vẫn sẽ không thể commit được mà bạn phải đưa nó về Staged cũng bằng lệnh `git add`.

Bỏ qua Staging Are để commit

Như mình có nói ở trên là một tập tin sau khi được thay đổi hay tạo mới thì nó phải được thêm vào Staging Area với lệnh git add. Tuy nhiên, bạn có thể đưa một tập tin đã được Tracked để commit mà không cần đưa nó vào Staging Area với tham số `-a` trong lệnh `git commit`. Ví dụ: `git commit -a -m "Skipped Staging Are to commit"`.

Tìm hiểu thêm về trạng thái



Vòng đợi trạng thái của các tập tin

Untracked

Nếu bạn tạo ra hoặc thêm vào một tập tin mới vào trong thư mục làm việc của bạn thì nó sẽ ở trạng thái Untracked. Bây giờ mình thử tạo ra một tập tin mới tên là *faq.html*, sau đó dùng lệnh `git status` để xem trạng thái của Git trong thư mục làm việc.

```
$ touch faq.html
```

```
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Untracked files:
 (use "git add <file>..." to include in what will be committed)

 faq.html

nothing added to commit but untracked files present (use "git add" to track)
```

Note: Lệnh `touch` là tạo ra một tập tin rỗng.

Bây giờ bạn sẽ thấy nó đã liệt kê ra tên tập tin đang ở trạng thái Untracked. Để đưa nó về Tracked bạn sẽ sử dụng lệnh `git add` và xem lại trạng thái của nó.

```
$ git add faq.html
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
 (use "git reset HEAD <file>..." to unstage)

 new file: faq.html
```

Bây giờ bạn thấy, tập tin faq.html của mình đã được đưa về trạng thái Staged và nó có thể được commit. Tại sao? Vì bạn phải biết rằng nếu một tập tin ở trạng thái Untracked mà được đưa về Tracked thì nó sẽ nằm ở

trạng thái Staged luôn, trừ khi bạn thay đổi nội dung tập tin này thì nó sẽ đưa về trạng thái Modified và nó không thể commit trừ khi bạn gõ lệnh `git add` cho nó.

Tracked

Một khi một tập tin đã được đưa về Tracked thì nó sẽ có thể thay đổi giữa 3 trạng thái khác nhau là **Modified**, **Unmodified** và **Staged**.

Trước hết bây giờ mình đã có một tập tin mới đã được đưa về Staged với lệnh `git add` như ví dụ trên. Bây giờ mình tiến hành thay đổi nội dung của tập tin faq.html này và xem kết quả của lệnh `git status`.

```
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
 (use "git reset HEAD <file>..." to unstage)

 new file: faq.html

Changes not staged for commit:
 (use "git add <file>..." to update what will be committed)
 (use "git checkout -- <file>..." to discard changes in working directory)

 modified: faq.html
```

Bạn có thấy sự kỳ lạ ở tập tin *faq.html* không? Đó là nó được hiển thị ở hai trạng thái Staged (có thể commit) và Modified (không thể commit) hay còn gọi là Unstaged. Sở dĩ có sự kỳ lạ đó ở đây là bởi vì trước đó bạn đã tạo ra tập tin *faq.html* và đưa về Tracked thì nó cũng đã được đưa về Staged để có thể commit. Tuy nhiên sau đó bạn lại chỉnh sửa nội dung của nó nên nó đã có một phiên bản khác nằm ở trạng thái Modified (không thể commit). Nếu bây giờ bạn gõ lệnh git commit để ủy thác nó thì bản chụp của tập tin faq.html ở lần cuối cùng bạn gõ lệnh git add sẽ được commit lên chứ nó không chứa các nội dung mà bạn vừa thêm vào. Và để nó có thể commit tập tin faq.html đã được chỉnh sửa thì bạn phải gõ lại lệnh `git add faq.html` lần nữa.

Chuyển tập tin từ Untracked về Tracked

Trong Git, bạn có thể đưa một tập tin từ Tracked về Untracked với lệnh rm tên_file. Lệnh rm sẽ giúp bạn đưa tập tin về trạng thái Untracked nhưng không xóa hẳn trong ổ cứng.

```
$ rm faq.html
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
```

```
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)
```

```
deleted: faq.html
```

Còn nếu bạn muốn xóa nó luôn thì dùng lệnh `git rm -f tên_file` và ***nhớ cẩn thận*** khi dùng lệnh này.

Lời kết

Có thể bạn sẽ thấy những gì mình nói trong bài này hơi dài nhưng đó là những kiến thức rất cơ bản về Git mà bạn cần nắm chắc vì nếu bạn không hiểu vòng đời các trang thái của một tập tin trong Git thì chắc chắn sau này khi làm việc bạn sẽ bối rối khi gõ lệnh git commit và cứ hỏi tại sao lại không commit được vì muốn commit được bạn sẽ phải đưa nó về trạng thái thích hợp, tức là trạng thái Staged.

Xem tiếp bài trong serie

Phần trước: [Git] Cách tạo một repository

Phần kế tiếp: [Git] Git Log và Undo Commit

Có liên quan

[Git] Git và Github là gì? Tại sao nên dùng?
21/04/2015
Trong "Công cụ"

[Git] Cách tạo một repository
23/04/2015
Trong "Công cụ"

[Git] Branch - Kỹ thuật phân nhánh
23/04/2015
Trong "Công cụ"

git linux

Thạch Phạm

Đam mê với web và lập trình, thích viết và chia sẻ, nghiện cà phê và xăm mình, hứng thú với nhạc dân ca và nhạc không lời.

Xem thêm bài viết Subscribe

Để lại bình luận

Hãy là người "bóc tem"

Thông báo cho trả lời cho bình luận đang theo dõi Email ›

b i link b-quote u img ul ol li code

Bắt đầu thảo luận

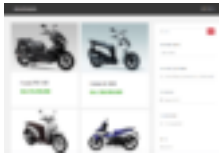
Bài mới nhất

Hãy tắt XML-RPC khi không cần sử dụng

18/09/2016

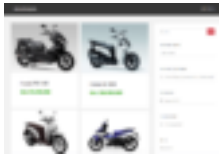
19 themes WordPress miễn phí đẹp nhất tháng 8 – 9/2016

18/09/2016



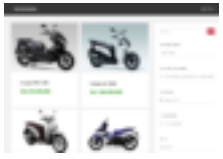
Làm trang shop đơn giản với WordPress – 07

04/09/2016



Làm trang shop đơn giản với WordPress – 06

04/09/2016



Làm trang shop đơn giản với WordPress – 05

31/08/2016

Dành cho người mới

- Học WordPress cơ bản
- Học làm website bán hàng
- Học sử dụng VPS/máy chủ Linux
- Học HTML cơ bản
- Học CSS cơ bản

Mới cập nhật

Hãy tắt XML-RPC khi không cần sử dụng

Cập nhật lúc 20/09/2016

19 themes WordPress miễn phí đẹp nhất tháng 8 – 9/2016

Cập nhật lúc 18/09/2016

LEMP cho VPS [Phần 1] – Cài đặt NGINX và PHP-FPM

Cập nhật lúc 17/09/2016

Giới thiệu

Tham khảo

THACHPHAM
SHARE THE BEST

ThachPham.Com là blog cá nhân tập trung vào việc phổ biến kiến thức WordPress nói riêng và các kiến thức làm website nói chung đến tất cả mọi người.
Phương châm hoạt động của thachpham.com đó là chia sẻ không giới hạn về những kiến thức bổ ích nhất liên quan đến quản trị website.

- **Các dịch vụ Shared Host tốt nhất**
- **Các dịch vụ VPS tốt nhất**
- **Nên mua theme WordPress ở đâu?**

Dịch vụ

- **Cài đặt WordPress miễn phí**
- **Cài đặt máy chủ**
- **Thiết kế website WordPress**
- **VPS SSD tốc độ cao**
- **Tư vấn miễn phí**

Liên kết

- Trần Ngọc Chính
- IZWebz
- Vietdesigner
- Danny Nguyễn
- Ngân Sơn
- Văn Mỹ
- Lập trình Sen Việt
- Trung Đức
- Vinacode
- Giúp Bạn Làm Đẹp

- Lại Văn Đức
- Ninh Đôn
- Đinh Trang
