Git Phổ Thông

Lê Quang Phúc, Lê Quang Phú

Ngày 24 tháng 3 năm 2013

Tóm tắt nội dung

Tài liệu này giới thiệu các lệnh thường dùng với Git thông qua các ví dụ cụ thể.

Mục lục

1 Giới thiệu

Sau một thời gian sử dụng Git trong công việc, bản thân tôi đã trải qua nhiều giai đoạn, từ làm quen với Git, cho đến đau khổ với Git, và bây giờ là yêu Git. Quyển sách này giới thiệu một số lệnh Git cơ bản được dùng trong một số tình huống cụ thể

2 Làm thế nào để cài đặt Git

Thiết lập hệ thống làm việc với Git trên Linux và Windows đều rất đơn giản. Trên Ubuntu, bạn cài đặt Git với **apt-get** như sau

Code 1: Cài đặt Git trên Ubuntu

```
# cac lenh linux
$ sudo apt-get install git-core      # cai dat git
```

Trên RedHat, bạn cài đặt Git với **yum** như sau

Code 2: Cài đặt Git trên Ubuntu

```
# cac lenh linux
$ sudo yum install git               # cai dat git
```

Với Windows, bạn dùng google và tìm kiếm với từ khóa **git bash**. Sau đó cài đặt git-bash vào là có thể dùng được Git.

Sau khi thiết lập được hệ thống Git trên máy tính của mình, để thực hành với Git các bạn có thể sử dụng GitHub để tạo repository của Git miễn phí. Sau khi đã có tài khoản ở GitHub thì chúng ta sẽ coi GitHub như là remote server chứa remote repository của chúng ta, và máy tính của chúng ta sẽ chứa local repository.

3 Sao chép dự án với Git

Để bắt đầu làm việc với một dự án có sẵn, chúng ta sao chép dự án đó về máy của mình với **git clone**.

Code 3: Sao chép

```
# cac lenh git
$ git clone git_url                    # sao chep
```

4 Làm quen với Git

Để có thể bắt đầu làm việc được với Git, chúng ta chỉ cần biết đến bốn câu lệnh , **status**, **add**, **commit**. Câu lệnh **git init** dùng để khởi tạo Git cho một dự án. Sau khi thực hiện lệnh này bạn có thể bắt đầu sử dụng các lệnh của Git để quản lý dự án của mình. Câu lệnh **git status** giúp bạn xem trạng thái của Git và tôi nghĩ rằng bạn sẽ thường xuyên muốn làm như vậy khi sử dụng Git. Để cập nhật những thay đổi trong dự án của mình với Git, bạn sử dụng câu lệnh **git add**. Cuối cùng, để lưu lại trạng thái của toàn bộ dự án tại một thời điểm thì bạn sẽ dùng câu lệnh **git commit**

Code 4: Bốn lệnh cơ bản

```
# cac lenh git
$ git init                             # khoi tao Git
$ git status                           # Xem trang thai
$ git add                              # luu chinh sua
$ git commit                           # luu trang thai du an
```

5 Thực chiến với bốn lệnh cơ bản của Git

5.1 Khởi tạo dự án

Trong tình huống mà bạn muốn áp dụng Git để quản lý một dự án của mình, bạn chuyển thư mục hiện thời của mình đến thư mục chứa dự án và sử dụng **git init**, là lệnh đầu tiên trong 4.

Code 5: Bốn lệnh cơ bản

```
# cac lenh he dieu hanh
```

```
$ cd du-an-voi-git                          # chuyen thu muc
$ git init                                  # Khoi tao git
$ git status                                # xem trang thai
$ cat > file_moi.txt                        # tao file
noi dung file 1
noi dung file 2
(ctrl + D)
$ cat file_moi.txt                          # xem lai noi dung
$ git status                                # file moi chua vao git
$ git add file_moi.txt                      # dua file vao git
$ git status                                # file da vao git
$ git commit                                # luu trang thai
```

6 Làm việc với branch

Khi làm việc với các dự án chúng ta thường phải phân nhánh tại một thời điểm nào đó, chẳng hạn để phân biệt bản chính với bản đang phát triển, hay để phát triển thêm nhứng tính năng mới. Với Git chúng ta có thể dễ dàng đạt được điều này.

6.1 Tạo nhánh mới

Code 6: Tạo nhánh

```
# cac lenh git
$ git branch                              # xem danh sach nhanh local
$ git branch -r                           # xem danh sach nhanh remote
$ git branch -a                           # xem tat ca cac nhanh
$ git branch ten_nhanh                    # tao nhanh moi
$ git push origin ten_nhanh:ten_nhanh# tao nhanh o remote
```

Thông thường sau khi tạo nhánh mới xong bạn sẽ muốn làm việc ngay trên nhánh vừa được tạo, khi đó bạn có thể thực hiện những lệnh sau

Code 7: Tạo nhánh tắt

```
# cac lenh git
$ git branch ten_nhanh                    # tao nhanh moi
$ git checkout ten_nhanh                  # chuyen nhanh lam viec
# hai lenh tren tuong duong voi lenh sau
$ git checkout -b ten_nhanh               # tao nhanh moi, chuyen nhanh
```

6.2 Xóa nhánh

Để xóa một nhánh chúng ta dùng các lệnh sau

Code 8: Xóa nhánh

```
# cac lenh git
$ git branch -d ten_nhanh              # xoa nhanh o local
$ git push origin :ten_nhanh           # xoa nhanh o remote
```

6.3 Ghép nhánh

Để ghép hai nhánh lại với nhau chúng ta sử dụng lệnh sau đây

Code 9: Ghép nhánh local

```
# cac lenh git
$ git checkout ten_nhanh             # chuyen nhanh
$ git merge nhanh_local              # ghep voi nhanh local khac
```

Code 10: Ghép nhánh remote

```
# cac lenh git
$ git checkout ten_nhanh             # chuyen nhanh
$ git merge remote:nhanh_remote      # ghep voi nhanh local khac
```

7 Một số câu hỏi thường gặp

7.1 Tạo và xóa tag ở remote server

Để tạo một tag ở remote server bạn sử dụng các lệnh sau đây.

Code 11: Tạo tag ở remote server

```
# cac lenh git
$ git tag                              # xem list cac tag hien co
$ git tag -a ten_tag -m 'ghi chu'      # khoi tao Git
$ git tag                              # xem lai list cac tag
$ git push origin ten_tag              # push len remote server
```

Tình huống ở đây là bạn đã tạo nhầm một tag ở remote server với các lệnh trong Code 11 và bạn muốn xóa nó đi.

Code 12: Xóa tag ở remote server

```
# cac lenh git
$ git tag                                # xem list cac tag hien co
$ git tag -d ten_tag                     # khoi tao Git
$ git tag                                # xem lai list cac tag
$ git push origin :refs/tags/ten_tag # push len remote server
```

7.2 Sử dụng cherry-pick để gắp các commit đơn lẻ

8 Cấu trúc dữ liệu của Git

Git lưu trữ dư liệu trên bôn cấu trúc dữ liệu chính, đó là:

- blob
- tree
- commit
- tag

8.1 Blob: đơn vị lưu trữ nhỏ nhất của Git

Nội dung của các file trong một dự án được quản lý bởi Git được lưu trữ trong các blob. Mỗi file sẽ tương ứng với một và chỉ một blob.

8.2 Tree: cấu trúc lưu trữ của Git

Nội dung của các file (được lưu bởi blob) sẽ là các nốt lá (nốt cuối của một nhánh) trên cây lưu trữ của Git. Các nhánh lại được liên kết với nhau tạo thành cây.

8.3 Commit: các cột mốc lịch sử

Mỗi commit lưu trữ thông tin về người thực hiện commit, một con trỏ chỉ đến một nút trên cây lưu trữ trạng thái của dự án, các con trỏ chỉ đến các nốt bố mẹ để dẫn đến trạng thái này.

8.4 Tag: sự kiện