BÁO CÁO PRE-INTERN NGÀY 30/11/2023

Tìm hiểu các trạng thái, ý nghĩa của các câu lệnh git:

$ git init: Khởi tạo repo mới

* Status:

$ git status: xem thông tin trạng thái sửa đổi, thêm mới, xóa các file trước khi thực hiện commit

$ git status –s: hiển thị thông tin ngắn gọn hơn, với các file có thể có các ký tự tương ứng với các thông tin gồm:

' ' = unmodified (không đổi)

M = modified (có sửa đổi)

A = added (file mới thêm)

D = deleted (file bị xóa)

R = renamed (đổi tên file)

C = copied (file copy từ file khác)

U = updated but unmerged (đã cập nhật, nhưng chưa merge)

* Branch:

$ git branch: hiển thị tất cả tên các nhánh

$ git branch newname: tạo một nhánh mới ( newname là tên nhánh mới tự đặt)

$ git checkout newname: chuyển sang nhánh newname

$ git checkout –b newname: tạo nhánh mới và chuyển sang nhánh vừa tạo

$ git branch –a: kiểm tra các nhánh có trên remote

* Add:

$ git add -FileName: lưu thông tin để chuẩn bị cho lệnh commit (FileName: tên file)

$ git add . : lưu toàn bộ thông tin

* Commit:

$ git commit -m "Ghi chú về commit": lưu dữ liệu vào hệ thống git

$ git commit –a –m "Ghi chú về commit": cho thêm tham số -a tương đương lệnh git add

$ git commit --amend -m "Thông tin về commit": trong trường hợp commit đã được tạo ra nhưng chưa thực hiện push thì có thể tạo commit mới thay thế cho commit cuối cùng đó. Dùng trong trường hợp không muốn tạo ra nhiều commit trong lịch sử.

* Reset:

$ git reset --soft HEAD~1: tham số --soft sẽ hủy commit cuối, con trỏ HEAD sẽ chuyển về commit cha. Đồng thời những thay đổi của commit cuối được chuyển vào vùng staging (vùng nội dung được lưu khi thực hiện lệnh git add) nhằm để có cơ hội commit lại hoặc sửa đổi.

Thay bằng tham số --hard kết quả giống tham số --soft, chỉ có một khác biệt là nội dung thay đổi của commit cuối không đưa đưa vào staging mà bị hủy luôn.

$ git reset: hủy thao tác của lệnh git add đã được dùng.

$ git reset – filename: hủy một file nào đó trong vùng staging chứ không phải toàn bộ

* Log:

$ git log: xem lại lịch sử các commit

Lệnh git log có nhiều tham số đề xuất ra, định dạng các thông tin theo cách mong muốn. Một số phím chức năng có thể dùng để điều hướng và tìm kiếm trong log như:

return - dòng tiếp theo

w - trang tiếp

spacebar - trang trước

q - thoát

?pattern - tìm kiếm, với pattern là mẫu tìm kiếm (keyword)

/pattern - giống ?pattern

n - đến vị trí tìm kiếm phía dưới

N - đến kết quả tìm kiếm phía trước

Một số thiết lập hay dùng với git log:

$ git log -2: Nếu chỉ muốn hiện thị một số commit log, ví dụ hiện thị log của 2 commit cuối thì cho thêm -2 vào lệnh

$ git log -p -2: Nếu muốn hiện thị chi tiết các thay đổi của từng commit thì thêm vào tham số -p

$ git log --stat -5: Nếu hiện thị thống kế gọn hơn về sự thay đổi thì dùng tham số --stat, hoặc dạng ngắn gọn hơn là --shortstat

Định dạng thông tin chung về commit (mã hash, dòng thông tin) trên một dòng thì dùng tham số --oneline:

$ git log --oneline

$ git log --stat -10 --oneline

Lọc kết quả với git log:

**Lọc theo ngày:** Có thể dùng tham số --after="year-month-day" hoặc --before="year-month-day" hoặc dùng cả hai để chỉ ra khoảng ngày. Ví dụ: hiện thị các log từ ngày 1/1/2019 đến ngày 31/12/2019

$ git log --after="2019-1-1" --before="2019-12-31"

**Lọc theo người commit** dùng tham số --author="tác giả", có thể kết hợp nhiều người bằng ký hiệu \|

$ git log --oneline --author="XuanDieu"

**Lọc theo thông tin ghi chú về commit** sử dụng thiết lập --grep="keyword ..."

$ git log --oneline --grep="updated"

**Lọc các commit liên quan đến file cụ thể**, sử dụng thiết lập -- rồi liệt kê các file

$ git log --oneline -- src/test.php

**Lọc theo nội dung cập nhật** sử dụng tham số -S"nội dung tìm"

$ git log --oneline --shortstat -S"sendmail"

**Lọc các commit bình thường** (tham số --no-merges) và các commit do gộp nhánh (tham số --merges)

$ git log --merges

* Push:

$ git push: sử dụng để đẩy các commit mới ở máy trạm (local repo) lên server (remote repo). Nguồn để đẩy lên là nhánh mà con trỏ HEAD đang trỏ tới.

$ git push -u origin master: Nếu là lần đầu tiên đẩy Local Repo lên Remote Repo mới khởi tạo thì cần tạo ra một theo dõi kết nối, upstream giữa local và remote, vậy hãy dùng tham số -u. Trong trường hợp này ví dụ tạo upstream cho nhánh master.

$ git push: Sau khi có upstream, mỗi lần cần đẩy dữ liệu lên remote của nhánh master, chỉ việc thực hiện lệnh git push.

$ git push origin newName: đẩy lên nhánh cụ thể (newName: tên nhánh)

$ git push origin –all: đẩy tất cả các nhánh ở local lên server

$ git push origin –delete newName: xóa nhánh trên remote có tên origin (newName: tên nhánh)

* Merge:

$ git merge: để gộp nhánh này vào nhánh khác. Ví dụ muốn merge nhánh newName vào master, checkout sang master và thực hiện lệnh: $ git merge newName

* Pull:

$ git pull

$ git pull origin : chỉ rõ remote

Git tải về thông tin từ remote và ngay lập tức merge cho nhánh đang làm việc của local repo.

* Stash:

$ git stash save

$ git stash: lưu lại các thay đổi chưa commit mà muốn đổi sang nhánh khác khi đang làm dở nhánh hiện tại

* Git ignore: liệt kê những file mà mình không mong muốn cho vào git hoặc hiểu nôm na là Git sẽ bỏ qua những file đó đi.

Các bước cần thực hiện:

B1: Tạo file có đuôi .gitignore trong thư mực repo

B2: liệt kê danh sách các file/ thư mục không muốn theo dõi trong file này

B3: git add để thêm file .gitignore vào danh sách staging.