Naming

Pasnal =>Chữ cái đầu viết hoa và chữ cái sau cũng viết hoa MyName => tên lớp bắt buộc theo kiểu này

ComeCCasel => chữ cái đầu viết thường chữ cái sau viết hoa myName => tên biến

Imder-score =>nối nhau bằng nối gạch dưới => my\_name => biến dùng trong cơ sở dữ liệu

Leebab-case => nối với nhau bằng gạch ngang nối gạch ngang => my-name => dùng trong css

Snake-case =>nối với nhau bằng gạch ngang nối gạch ngang => my-name

Git

VCS là hệ thống quản lý phiên bản

+ Quản lý thay đổi mã nguồn

+ giải quyết vấn đề xung đột

VCS 2 loại

+VCS tập chung

Nhược điểm nếu máy tính ko có mạng gặp vấn dề thì lập trình viên sẽ ko thể lưu chữ

+ VCS nối tiếp

Khác phục mọi vấn đề của vcs tập chung

Mỗi máy tính được coi là 1 kho lưu chữ do dó kể cả ko có mạng vẫn có thể lưu trữ

Đại diện là git

Git

-là một vcs phân tán

Mã nguồn Linux

Linux (phát triển châu âu) khác untx(mỹ)

Linux androi

Untx Mac Os ,ios

- phổ biến nhất thế giới

- git khác github

Git là một phần mềm / một giao thức Quản lý mã nguồn

GitHub công ty cung cấp lưu trữ tập chung cho git

Cài đặt git

Cách làm việc

+ sử dụng giao diện dòng lenehk để giao tiếp với git

+ cách 2: sử dụng phần mềm giao tiếp với git

Phân biệt

CLI sử dụng toàn bộ tính năng của it, hiểu sâu về git

GNI không đầy đủ tính năng dễ dùng

Kiến trúc

Gồm local và remote

Local:

+working diretory : thư mộc dự của mình làm việc

+ Staging area : Vùng dệm để chuẩn bị sao lưu

+ local repo: nới lưu trữ sao lưu

Staging area ,local repo nẳm trong working diretory

Git init =>tạo ra 1 forder git

.git

-Staging area

Trước khi

Git add <tên file>

Git .

Remote Repo

Là nơi lưu trữ máy chủ

Snapshot ảnh chụp thay đổi của mã nguồn

-Trạng thái của 1 file

Kiểm tra trạng thái

git status

để sử dụng lệnh commit dùng lệnh sau

git commit -–m<” comment” >

-head dạng như một con trỏ chỉ vị trí hiện tại trong dòng thời gian của dự án. Có thể sử dụng để di chuyển qua ại giữa các commit để xem mã nguồn tài tại mỗi thời điểm

Để có thể di chuyển giữa các commit trong timeline, sử dụng lệnh sau

Git checkout<commit>

-Xem lịch sử commit

Git log

Phím w trang tiếp theo

Phím Q thoát cái log đó

Return xuống dòng tiếp theo

Phím lên xuống để lên xuống dùng tiếp theo

Để tìm kiếm trong log

- git grep

Quy trình

Tạo dự án => git init => thêm mới hoặc sửa đổi tập tin thư mục dự án.

=>Đưa tập tin hoặc thư mục vào staging area thông qua lệnh git add

=>Lưu lại snapshot thay đổi của dự án vào CSDL của git bằng lệnh Git commit-m”note”

=>lặp lại từ bước 3

=> kiểm tra lịch sử thay đổi bằn git log

-Để mã nguồn đến remote repo, Sử dụng lệnh sau

Git remote <tên\_remote\_repo>,<link\_remote\_repo>

-Để kiểm tra xem local kết nối đến bao nhiêu remote repo

-git remote -v

-Sau khi kết nối với remote repo lập trình viên có thể tiến hành đẩy code và commit lên remote repo với câu lẹnh sau

Git push<tên\_remote\_repo><tên nhánh>

Lập trình viên kéo code bằng cách

+ git full

Lập trình viên về mà ko cần đồng bộ

+ git fetch

Dùng để xem có thay đôi j mới của thư mục ko

- khi muốn tham gia vào dự án

Git clone<link>

-Quy trình

1 tạo dự án hoặc clone dự án về từ Remote Repo

2 Tạo repo cho dự án bằng lệnh git init (trong trường hợp không clone từ remote repo)

3 Thêm mới tập tin hoặc sử đổi tập tin thư mục trong dự án.

4. đưa thư muc vào stagging area thông qua lệnh git add <file thay đổi> hoặc git add .

5. Lưu snapshot thay đổi của dự án CSDL của git bằng lệnh git commit

6. đẩy hoặc kéo code về

7 bước 3

8 kiểm tra lại lịch sử bằng git log

.

Branch(nhánh)

Lệnh

Git breanch

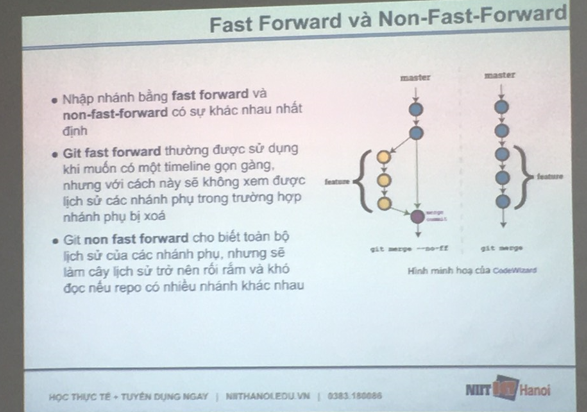
|  |  |
| --- | --- |
| Lệnh | Ý nghía |
| Git branch | Liệt kê các nhánh trong Repo |
| Git branch <tên nhánh> | Tạo nhánh mới trong local Repo |
| Git checkout<tên nhánh> | Di chuyển đến nhánh <tên nhánh> từ nhánh hiện tại trong local repo |
| Git checkout<tên nhánh> -b | Tạo nhánh mới tên là <tên nhántrong local Repoh> và di chuyển đến <tên nhánh> |
| Git branch -D<tên nhánh> | Xóa nhánh <tên\_nhánh>, Cần di chuyển sang nhánh khác trước khi xóa |
| Git push<tên remote repo>  <tên nhánh trên remote repo > | Đẩy mã nguồn và comit lên remote repo tại  Tên nhánh trên remote repo |
| Git branhch –r | Xem danh sách các branch trên remote repo |

Nhập nhánh

Smerg

- no-ff luôn sinh ra commit tại điểm nhập nhánh

- ff lấy toàn bộ commit từ nhánh phụ

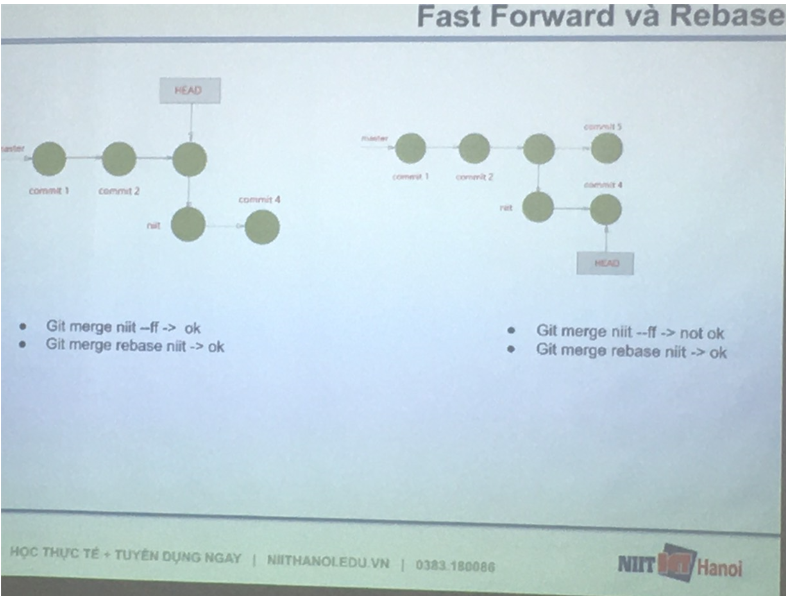


Rebase

Cũng là phân nhánh

Nếu gitignore bị lỗi bị cached

-git rm –r –cached .



\* giải quyết xung đột khi nhập nhánh

\*cherry Pick

Git cherry-pick <hash commit id> nhập nhánh

Git cherry-pick –continue khi nỗi tiếp nhập xong khi sử lý nnoiox

\*git flow

Master: được tạo ra khi tạo project

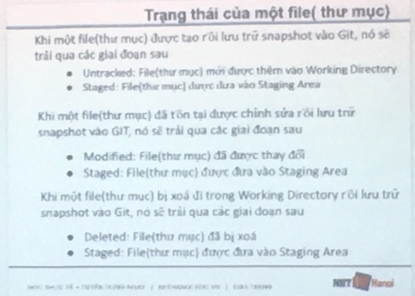
Develop: ae với nhóm master để phát triển ko được xóa

Feature: Sinh ra khi phát triển tính năng mới. Sẽ bị xóa sau khi phát triển tính năng . Đặt tên(feature/tên tính năng)

Sinh ra từ develop vì bị xóa khi nhập vào develop

Release branch Sinh ra để test. Sau khi test thì chuyển vào nhánh develop. Có thể xóa khi test xong

Hotfix: fix nóng . sinh ra từ nhóm master để fix lỗi . khi phí xong nhập vào nhóm master và develop.



Squash là một kỹ thuật cho phép nén toàn bộ commit thanhd một commit trước khi merge vào nhánh chính

Về cơ bản squash cũng khá giống việc sử dụng smerg