Tìm Hiểu Về Git Và SVN

**I.Git**

1. Git là gì ?

Git là một hệ thống quản lý phiên bản phân tán ( Distribited Version Control System - DVCS ), nó là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay.

Git cung cấp cho mỗi lập trình viên kho lưu trữ ( respository ) riêng chứa toàn bộ lịch sử thay đổi.

1. Version Control System – VCS là gì và tác dụng của VCS.

VCS là viết tắt của **Version Control System** là **hệ thống kiểm soát các phiên bản phân tán mã nguồn mở**. Các VCS sẽ lưu trữ tất cả các file trong toàn bộ dự án và ghi lại toàn bộ lịch sử thay đổi của file. Mỗi sự thay đổi được lưu lại sẽ được vàthành một version (phiên bản).

VCS nghĩa là hệ thống giúp lập trình viên có thể lưu trữ nhiều phiên bản khác nhau của một mã nguồn được nhân bản (**clone**) từ một kho chứa mã nguồn (**repository**), mỗi thay đổi vào mã nguồn trên local sẽ có thể ủy thác (**commit**) rồi đưa lên server nơi đặt kho chứa chính.

Và một máy tính khác nếu họ có quyền truy cập cũng có thể clone lại mã nguồn từ kho chứa hoặc clone lại một tập hợp các thay đổi mới nhất trên máy tính kia.

Lập trình viên có thể xem lại danh sách các sự thay đổi của file như xem một dòng thời gian của các phiên bản. Mỗi phiên bản bao gồm: nội dung file bị thay đổi, ngày giờ sửa đổi, người thay đổi là ai, lý do thay đổi hay tên phiên bản…

* Tác dụng của Git :
* Lưu lại lịch sử các version của bất kỳ thay đổi nào của dự án. Giúp xem lại các sự thay đổi hoặc khôi phục (revert) lại sau này.
* Việc chia sẻ code trở nên dễ dàng hơn, lập trình viên có thể để public cho bất kỳ ai, hoặc private chỉ cho một số người có thẩm quyền có thể truy cập và lấy code về.

Vốn là một VCS nên Git cũng ghi nhớ lại toàn bộ lịch sử thay đổi của source code trong dự án. Lập trình sửa file, thêm dòng code tại đâu, xóa dòng code ở hàng nào…đều được Git ghi nhận và lưu trữ lại.

1. Git hoạt động như thế nào ?

Sự khác biệt chính giữa Git và bất kỳ VCS nào khác (bao gồm Subversion…) là cách Git nghĩ về dữ liệu của nó.

Về mặt khái niệm, hầu hết các hệ thống khác đều lưu trữ thông tin dưới dạng danh sách các thay đổi dựa trên file. Các hệ thống này (CVS, Subversion, Perforce, Bazaar, v.v.) coi thông tin chúng lưu giữ dưới dạng một tập hợp các file và những thay đổi được thực hiện đối với mỗi file theo thời gian.



Git không nghĩ đến hoặc lưu trữ dữ liệu của mình theo cách này. Thay vào đó, Git coi thông tin được lưu trữ là một tập hợp các snapshot – ảnh chụp toàn bộ nội dung tất cả các file tại thời điểm.

Mỗi khi bạn “commit”, Git sẽ “chụp” và tạo ra một snapshot cùng một tham chiếu tới snapshot đó. Để hiệu quả, nếu các tệp không thay đổi, Git sẽ không lưu trữ lại file — chỉ là một liên kết đến tệp giống file trước đó mà nó đã lưu trữ. Git nghĩ về dữ liệu của nó giống như dưới đây:



Đây là điểm khác biệt quan trọng giữa Git và gần như tất cả các VCS khác. Nó khiến Git phải xem xét lại hầu hết mọi khía cạnh của kiểm soát phiên bản mà hầu hết các hệ thống khác đã sao chép từ thế hệ trước. Điều này làm cho Git giống như một hệ thống tệp nhỏ với một số công cụ cực kỳ mạnh mẽ được xây dựng trên nó, thay vì chỉ đơn giản là một VCS.

1. Lợi ích của Git

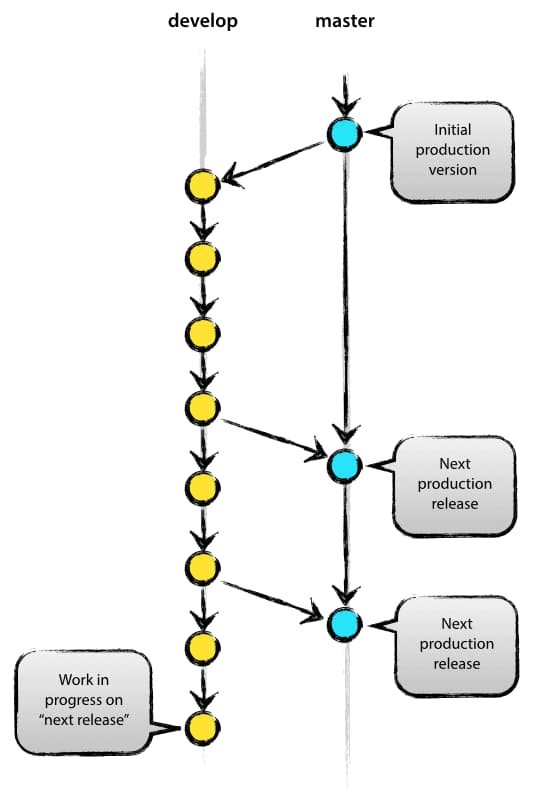
* Dễ sử dụng, thao tác nhanh, gọn, lẹ và rất an toàn.
* Sễ dàng kết hợp các phân nhánh (branch), có thể giúp quy trình làm việc code theo nhóm đơn giản hơn rất nhiều.
* Chỉ cần clone mã nguồn từ kho chứa hoặc clone một phiên bản thay đổi nào đó từ kho chứa, hoặc một nhánh nào đó từ kho chứa là bạn có thể làm việc ở mọi lúc mọi nơi.
* Deployment sản phẩm của bạn một cách không thể nào dễ dàng hơn.

1. Các thuật ngữ Git quan trọng

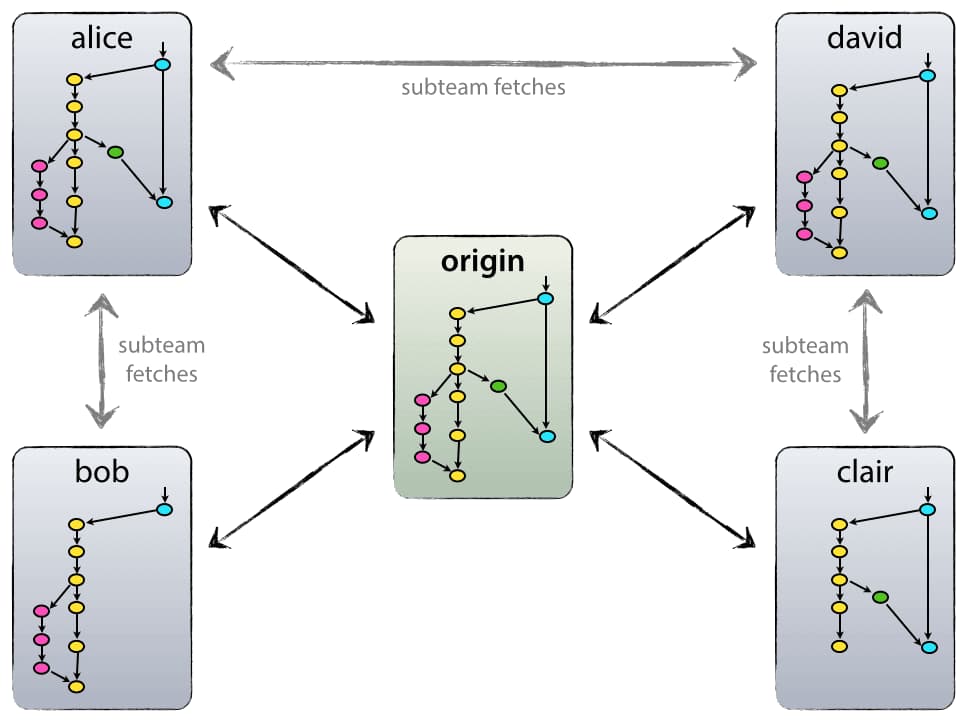
* Branch: Các **Branch**(nhánh) đại diện cho các **phiên bản cụ thể** của một kho lưu trữ tách ra từ project chính của bạn.

Branch cho phép bạn theo dõi các thay đổi thử nghiệm bạn thực hiện đối với kho lưu trữ và có thể hoàn nguyên về các phiên bản cũ hơn.

* Commit: Một commit đại diện cho một thời điểm cụ thể trong lịch sử dự án của bạn. Sử dụng lệnh commit kết hợp với lệnh **git add** để cho git biết những thay đổi bạn muốn lưu vào local repository.
* Checkout: Sử dụng lệnh git checkout để chuyển giữa các branch.
* Fetch: lệnh git fetch tìm nạp các bản sao và tải xuống tất cả các tệp branch vào máy tính của bạn
* Fork: Một fork là một bản sao của một kho lưu trữ (repository)
* Head: Các commit ở đầu của một branch được gọi là head
* Index: Bất cứ khi nào bạn thêm, xóa hoặc thay đổi một file, nó vẫn nằm trong chỉ mục cho đến khi bạn sẵn sàng commit các thay đổi.
* Master: là nhánh chính của tất cả các repository của bạn. Nó nên bao gồm những thay đổi và commit gần đây nhất



* Merge: Lệnh git merge kết hợp với các yêu cầu kéo (pull requests) để thêm các thay đổi từ nhánh này sang nhánh khác.
* Origin: Origin là phiên bản mặc định của repository. Lệnh git push origin master để đẩy các thay đổi cục bộ đến nhánh chính.



* Pull: Lệnh git pull được sử dụng để thêm các thay đổi vào nhánh chính
* Push: Lệnh git push được sử dụng để cập nhật các nhánh từ xa với những thay đổi mới nhất mà bạn đã commit.
* Rebase: Lệnh git rebase cho phép bạn phân tách, di chuyển hoặc thoát khỏi các commit.
* Remote: Một Remote (kho lưu trữ từ xa) là một bản sao của một chi nhánh.
* Repository: Kho lưu trữ Git chứa tất cả các tệp dự án của bạn bao gồm các branch, tags và commit.
* Stash: Lệnh git stash sẽ loại bỏ các thay đổi khỏi chỉ mục của bạn và xóa stashes chúng đi sau.
* Tags: Tags cung cấp cho bạn một cách để theo dõi các commit quan trọng.
* Upstream: Trong ngữ cảnh của Git, upstream đề cập đến nơi bạn push các thay đổi của mình, thường là nhánh chính (master branch).

1. Các lệnh Git cơ bản

* Git config: Để kiểm tra tên và kiểu email trong cấu hình dùng ***git config -- global user.name*** và ***git config -- global user.email***.
* Get init : Khởi tạo 1 git repository 1 project mới hoặc đã có. ***git init***
* Git clone: Copy 1 git repository từ remote source. ***git clone <:clone git url:>***
* Git status: Để check trạng thái của những file bạn đã thay đổi trong thư mục làm việc. ***git status***
* Git add: Thêm thay đổi đến stage / index trong thư mục làm việc. ***git add.***
* Git commit: commit nghĩa là một action để Git lưu lại một snapshot của các sự thay đổi trong thư mục làm việc.  ***git commit -m ”Đây là message, bạn dùng để note những thay đổi để sau này dễ dò lại”***
* Git push/ git pull:  Push hoặc Pull các thay đổi đến remote. Nếu bạn đã added và committed các thay đổi và bạn muốn đẩy nó lên hoặc remote của bạn đã update và bạn apply tất cả thay đổi đó trên code của mình. ***git pull <:remote:> <:branch:>*** and ***git push <:remote:> <:branch:>***
* Git branch: liệt kê tất cả các branch (nhánh). ***git branch*** hoặc ***git branch -a***
* Git checkout:  Chuyển sang branch khác. ***git checkout <: branch:>*** hoặc ***\*\* \_ git checkout -b <: branch:>*** nếu bạn muốn tạo và chuyển sang một chi nhánh mới.
* Git stash:  Lưu thay đổi mà bạn không muốn commit ngay lập tức. ***git stash***
* Git merge:  Merge 2 branch lại với nhau. Chuyển tới branch bạn muốn merge rồi  dùng ***git merge <:branch\_ban\_muon\_merge:>***
* Git reset: Bạn đã đưa một tập tin nào đó vào Staging Area nhưng bây giờ bạn muốn loại bỏ nó ra khỏi đây để không phải bị commit theo. ***git reset HEAD tên\_file***
* Git remote:  Để check remote/source bạn có hoặc add thêm remote. ***git remote*** để kiểm tra và liệt kê. Và ***git remote add <: remote\_url:>*** để thêm.
* git add: Để đưa một tập tin vào Staging Area. ***git add tên\_file*** hoặc muốn thêm hết file của thư mục thì ***git add all***

**II.SVN**

* 1. **svn là gì ?**

SVN là viết tắt của Subversion – là một hệ thống quản lý các version, Subversion là một nơi để các nhà phát triển phần mềm tạo mã lập trình có thể truy cập được cho mọi người. sau đó họ có thể thực thi khi họ muốn với mã, bao gồm ứng dụng để tải lên các phiên bản hiện tại hoặc sửa đổi mã.

Subversion (SVN) sử dụng một cơ sở dữ liệu - database trung tâm bao gồm mọi thứ mà phiên bản tập của bạn được quản lý với lịch sử đầy đủ. Các cơ sở dữ liệu cũng được công nhận là kho lưu trữ.

Kho lưu trữ thường tồn tại trên một máy chủ tập hoạt động chương trình máy chủ SVN cung cấp nội dung cho các máy khách SVN như TortoiseSVN theo yêu cầu.

* 1. **Cách thức hoạt động và những tính năng chính của SVN**

Subversion ban đầu được thiết kế giống như giao diện dòng lệnh. Người dùng chỉ cần mở Terminal sau đó gõ các lệnh văn bản là được. Để SVN hoạt động, người dùng cần thiết lập nó dựa vào 2 yếu tố chính:

* ***Yếu tố 1***: Máy chủ có tất cả phiên bản của tệp nguồn
* ***Yếu tố 2***: Bản sao cục bộ của các tệp trên máy tính của người dùng
* Các tệp trên máy tính người dùng được gọi là tệp tin làm việc. Đây là những tệp mà người dùng thực hiện việc chỉnh sửa và cam kết thay đổi với máy chủ SVN. SVN sẽ quản lý và ghi lại khi người dùng chỉnh sửa, đồng thời tạo ra một phiên bản mới. Người dùng thường làm việc với phiên bản mới nhưng nếu cần đến phiên bản cũ họ vẫn có thể quay trở lại.

**\*Những tính năng chính của svn :**

+ **SVN Checkout**: Sử dụng lần đầu tiên để kết nối với server chứa mã nguồn theo địa chỉ cung cấp.

+ **SVN Update**: Cập nhật tài nguyên mới nhất từ server về máy tính của từng thành viên trong nhóm.  
+ **SVN Commit**: Người lập trình sử dụng để đưa tài nguyên lên server khi được yêu cầu.

+ **Change Password**: Trong trường hợp máy tính của các bạn sử dụng nhiều tài khoản khác nhau với các dự án khác nhau. Nếu đã lưu mật khẩu hiện tại thì việc chuyển sang tài nguyên khác sẽ bị báo lỗi không truy cập do nó vẫn nhận tài khoản lưu nhớ. Để thay đổi các bạn chuột phải vào thư mục sau chọn TortoiseSVN => Settings => Chọn mục Saved Data => Nhấn nút xóa Clear All trong mục Authentication data trên giao diện hiển thị.

+ **Resolve**: Là chức năng cho phép bạn xử lý sung đột, sai khác giữa các phiên bản code mới nhau. Khi nó bị xung đột sẽ có biểu tượng dấm chấm than (!) màu vàng. Để thực hiện chức năng các bạn nhấn chọn TortoiseSVN => Resolve.

**+ Revert:** Là chức năng loại bỏ sự thay đổi trong mã nguồn hiện tại so với bản mới nhất đang lưu trữ trên server.

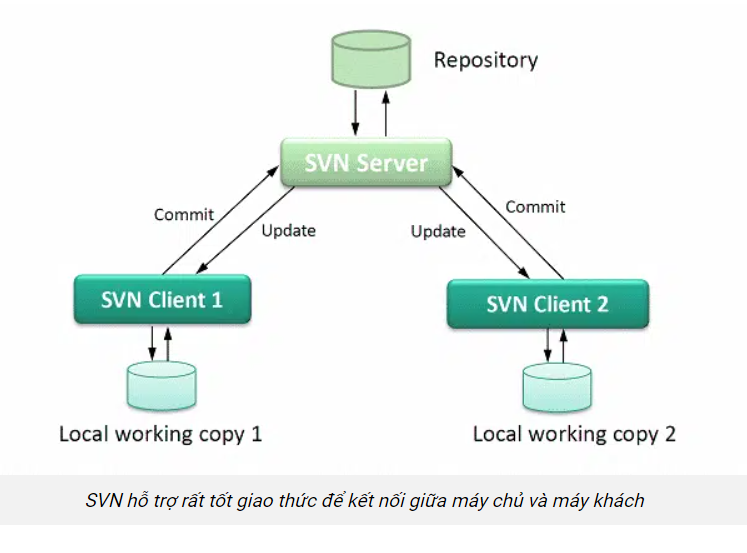
+ **Clean up**: Đây là chức năng được sử dụng khá nhiều khi có sự xung đột về code, không thực hiện được các chức năng Update, Commit trong SVN. Người dùng sẽ sử dụng chức năng này để loại bỏ các xung đột (confilict) trong code của dự án. Để sử dụng chức năng này các bạn chuột phải chọn TortoiseSVN => Clean up => Sau đó tích chọn các thông tin để xử lý trên giao diện hiển thị.

1. **Ưu điểm và lợi ích của SVN**

Với SVN bạn sẽ được hưởng các tính năng ưu việt và cũng chính là những ưu điểm của hệ thống quản lý version – SVN hiện này như sau:

* phiên bản được thực hiện cho các thư mục cũng bao gồm các tệp riêng lẻ.
* có một mục tiêu chuẩn để sao chép, xóa, thêm và đổi tên các tập tin trên các thư mục.
* thuộc tỉnh tập và thực mực được đăng ký theo thời gian, theo đó hỗ trợ bạn xây dựng một lịch sử toàn diện.
* giúp các nhà phát triển có thể thiết kế và dành các sửa đổi trong một phút, các đơn vị tự kiểm soát.
* SVN đơn giản để thực hiện các chức năng mạng hiện tại. Thứ sáu, tệp nhị phân và văn bản được lưu và quản lý theo cách tương thích.
* chi nhánh vía thẻ được xây dựng hiệu quả và kịp thời
* SVN rất đơn giản để được sử dụng bởi các ngôn ngữ lập trình và thiết kế các phần mềm khác nhau.

**\*Lợi ích của SVN**

****

* Subversion cung cấp chức năng để các thành viên trong nhóm có thể chỉnh sửa nội dung file một cách an toàn, đơn giản và nhanh chóng.
* SVN có khả năng thay thế hệ thống quản lý phiên bản CVS cũ (tồn tại nhiều nhược điểm). SVN có thể quản lý bất kỳ hệ thống phiên bản nào, được coi là hệ thống quản lý source code tập trung, mạnh mẽ, linh hoạt và hữu dụng.
* SVN quản lý thư mục và tập tin theo thời gian. Nó giống như hệ thống file server mà máy khách có thể download file một cách bình thường. Đặc biệt, SVN có khả năng lưu lại tất cả những thay đổi trên hệ thống file (ai thay đổi, thay đổi lúc nào, thay đổi những gì).
* SVN cho phép khôi phục lại những version cũ một cách nhanh chóng và chính xác. Do đó, làm việc nhóm cũng an toàn và hiệu quả hơn.
* Máy chủ và máy khách có thể cùng chạy trên một SVN, có nhiệm vụ ghi lại lịch sử dự án.
* SVN hỗ trợ rất tốt giao thức để kết nối giữa máy chủ và máy khách. Việc cho phép máy chủ hỗ trợ giao thức nào còn tùy thuộc vào cấu hình.

1. **Các phiên bản của SVN**

* ***Phiên bản SVN 1.1***: Lưu trữ tệp phẳng cho kho lưu trữ, tự động thoát URL, đặt các liên kết tượng trưng,…
* ***Phiên bản SVN 1.2***: Tạo khóa máy chủ trên các tệp, thực hiện tuần tự cam kết truy cập vào một số tài nguyên nhất định
* ***Phiên bản SVN 1.3***: Dựa trên đường dẫn đến SVN server để mang các điều kiện ủy quyền
* ***Phiên bản SVN 1.4***: Hỗ trợ sao chép kho dữ liệu một chiều
* ***Phiên bản SVN 1.5***: Tính năng theo dõi bán tự động của việc hợp nhất và phân nhánh, giải quyết xung đột tệp, kiểm tra và quản lý thay đổi từ phía máy khách,…
* ***Phiên bản SVN 1.6***: Tiếp tục làm cho việc hợp nhất và phân nhánh mạnh mẽ và hiệu quả hơn