GITHUB SHARING (PART IV): CÂU LỆNH THƯỜNG GẶP

DONG TX

[ADD COMMENT](http://csharpdepth.blogspot.com/2015/01/huong-dan-su-dung-github-part-3_10.html#comments)

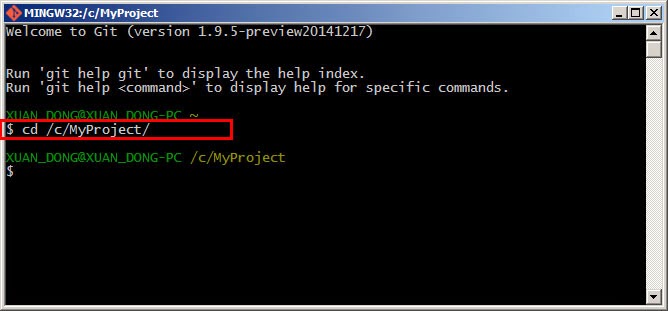
[GIT HUB](http://csharpdepth.blogspot.com/search/label/Git%20hub), [QUẢN LÝ SOURCE CODE](http://csharpdepth.blogspot.com/search/label/Qu%E1%BA%A3n%20l%C3%BD%20source%20code)

THỨ BẢY, NGÀY 10 THÁNG 1 NĂM 2015

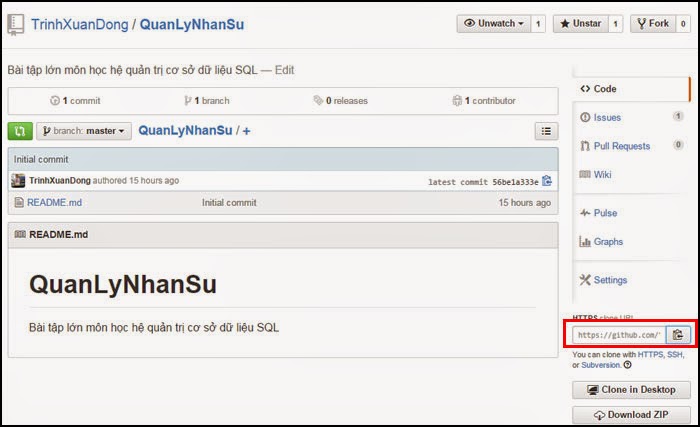
[](https://images-blogger-opensocial.googleusercontent.com/gadgets/proxy?url=http%3A%2F%2F3.bp.blogspot.com%2F-7XAvpTl8_TI%2FVLCX1q5UlfI%2FAAAAAAAAAe4%2FgrmXk9LvOck%2Fs1600%2FGithub_Par4.jpg&container=blogger&gadget=a&rewriteMime=image%2F*)*Git có nhiều câu lệnh, từ đơn giản đến phức tạp. Tuy nhiên khi mới bắt đầu sử dụng git mình nghĩ chỉ cần 1 số lệnh cơ bản. Ngoài ra git luôn hỗ trợ help mọi lúc (git help), chính vì vậy các bạn có thể tìm được cú pháp câu lệnh phù hợp với yêu cầu của mình. Bài viết này mình sẽ đề cập đến các câu lệnh git cơ bản nhất mà chúng ta thường xuyên phải sử dụng:*

**1 - Lấy source code trên server về máy local.**

Giả sử bạn muốn project được lưu trữ tại: c:\MyProject\, bước đầu tiên là mở Git bash lên và di chuyển thư mục hiện hành đến thư mục lưu trữ project bằng câu lệnh cd thông thường. Ở đây chúng ta sẽ sử dụng: cd /c/Myproject/

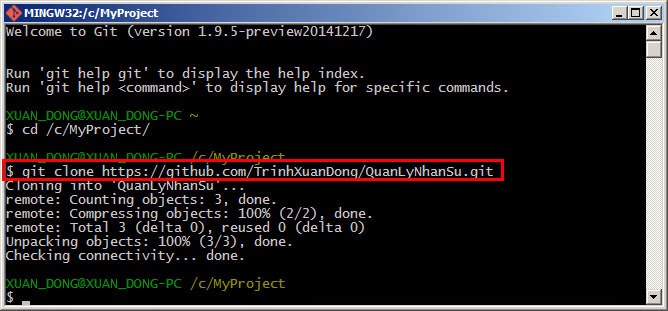
[](http://2.bp.blogspot.com/-jWeKpWGzsug/VLDHOhuVCeI/AAAAAAAAAhU/7cCdR6mGzgo/s1600/git_clone_1.jpg)

Bước tiếp theo đó là đăng nhập vào tài khoản git của bạn trên server và chọn project bạn muốn lấy về. Ở đây như bài trước, mình có 1 project là "QuanLyNhanSu". Click copy đường link như hình khoanh đỏ bên dưới:

[](http://1.bp.blogspot.com/-elXQMvQy_dg/VLDIX4NKL8I/AAAAAAAAAho/JvX2P_9sL2o/s1600/git_clone_2.jpg)

Đường link ở đây của mình sẽ là : <https://github.com/TrinhXuanDong/QuanLyNhanSu.git> . Và sử dụng câu lệnh:

git clone https://github.com/TrinhXuanDong/QuanLyNhanSu.git

[](http://4.bp.blogspot.com/-I16D7HFs4CI/VLDHOlKIQpI/AAAAAAAAAhQ/ZmfwgxlH59k/s1600/git_clone_3.jpg)

Nếu việc clone này là thành công. Trong thư mục MyProject của bạn sẽ có thư mục QuanLyNhanSu cùng với các file tương tự như trên git server. Việc này chẳng khác gì bạn có 1 đường link và bạn đi download file theo đường link đó.

**2 - Thêm file mới.**

Trước khi đọc tiếp các phần bên dưới, có một số lưu ý cũng như khái niệm mà bạn cần phải nắm rõ. Thứ nhất, toàn bộ những thay đổi dù chỉ là nhỏ nhất trong thư mục "QuanLyNhanSu" sẽ được git phát hiện ra, ví dụ như bạn thêm 1 file mới, hay sửa nội dung 1 file. Vấn đề ở đây là khi sửa nội dung 1 file thì đồng nghĩa với việc là file đó đã có sẵn khi bạn lấy dữ liệu từ trên git server về máy tính local. Ngược lại, khi bạn thêm 1 file thì đồng nghĩa với việc file đó trước đó ko có, và trên git server cũng ko có file này.

Mọi việc chúng ta làm mục đích duy nhất là đồng bộ được nội dung trên máy local và nội dung trên git server. Khi sửa nội dung 1 file và ta muốn nó đồng bộ với git server thì ta phải làm lần lượt các câu lệnh sau: commit ->push

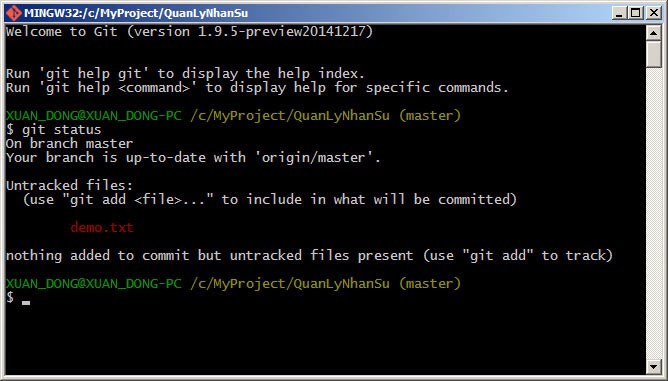
* Commit : Xác nhận việc thay đổi và đưa ra 1 thông điệp đi kèm (sẽ ví dụ bên dưới)
* Push : Đẩy toàn bộ nội dung trên local lên server

Khi thêm 1 file mới, quy trình sẽ là như sau: add ->commit->push

* Add: Xác nhận thêm file đó vào git. Nếu ko dùng add thì file đó sẽ ko bao giờ được đưa lên server.
* Commit : Nội dung như  phần sửa file ở trên
* Push : Nội dung như phần sửa file ở trên

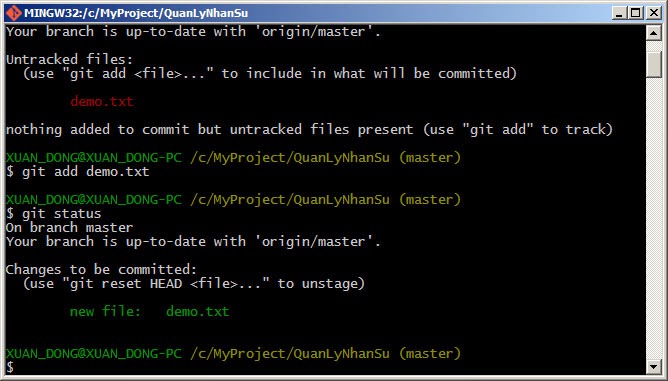
Ok, tiếp tục nào. Trong thư mục QuanLyNhanSu, bạn thêm 1 file mới. Giả sử ở đây mình thêm file demo.txt , sử dụng câu lệnh sau để biết được sự thay đổi:

git status

[](http://1.bp.blogspot.com/-G3ilHjEtUuk/VLaKPEh95AI/AAAAAAAAAkI/KK74_soGuwo/s1600/fit%2Bstatus.jpg)

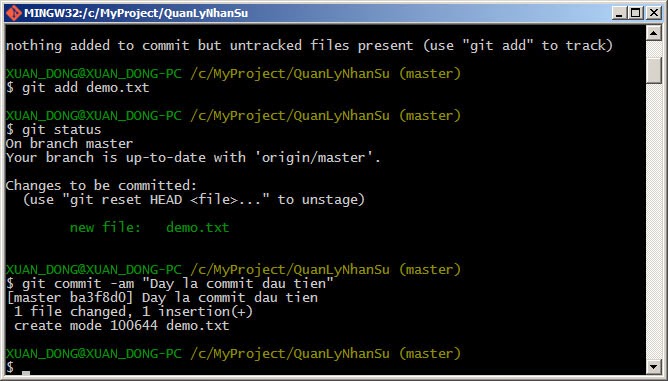
Sau khi gõ lệnh đó xong, bạn sẽ thấy git thông báo có file mới, và đánh dấu màu đỏ file đó. Dùng lệnh sau để báo với git rằng mình đồng ý thêm file này:

git add demo.txt

[](http://1.bp.blogspot.com/-QavLL5VNKmg/VLaKPKr7OXI/AAAAAAAAAkQ/kiDjjpGQI0o/s1600/git%2Badd.jpg)

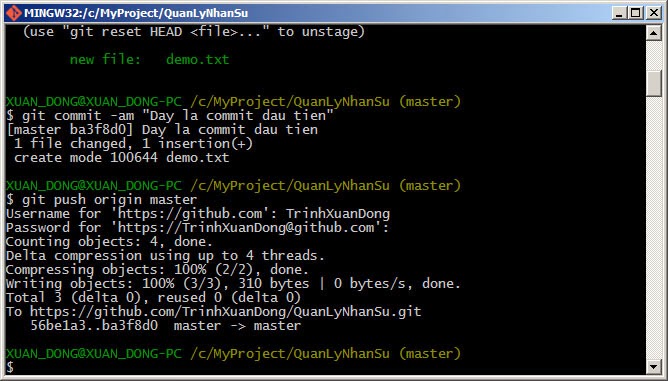
Tiếp tục sử dụng "git status" => Bạn sẽ thấy file demo.txt đã được chuyển màu xanh, điều này đảm bảo rằng file đó sẽ được đưa lên git server. Dùng tiếp câu lệnh sau:

git commit -am "<Nội dung commit>"

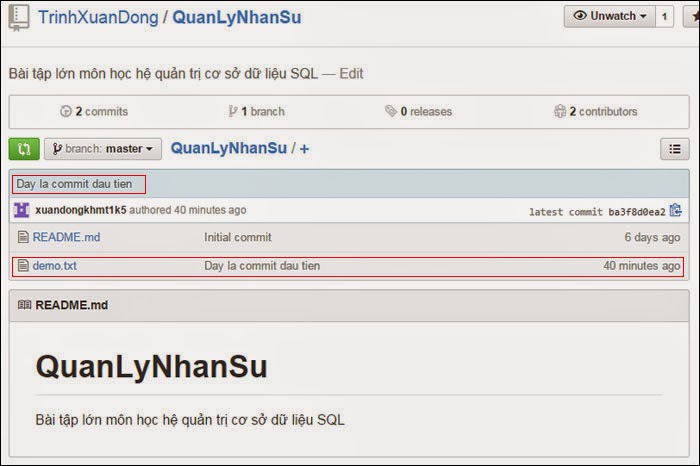
[](http://4.bp.blogspot.com/-GcGo8jokPNc/VLaKPDfpqtI/AAAAAAAAAkM/wn5E5igqFZY/s1600/git%2Bcommit.jpg)

câu lệnh này xác nhận **những file đã sửa đổi và những file bạn đã add ở bước trên** sẽ được đưa lên git server. Tiếp tục dùng lệnh sau để đưa toàn bộ dữ liệu lên server. Nhớ nhập tên tài khoản và mật khẩu nhé:

git push origin master

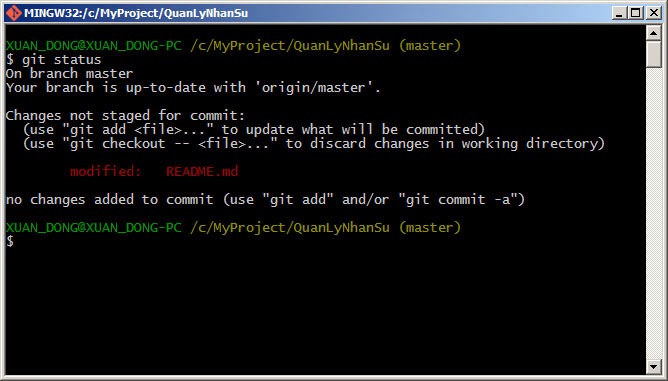
[](http://1.bp.blogspot.com/-y2KoSrTdAWc/VLaKQ4M-A2I/AAAAAAAAAkc/Id5ZgTlWXw0/s1600/git%2Bpush.jpg)

Master ở đây có nghĩa là nhánh chính, git làm việc theo nhánh, có 1 nhánh chính xuyên suốt trong quá trình bạn làm project, khi bạn sửa hoặc thêm file thì bạn đang làm ở nhánh phụ. Điều này đảm bảo rằng nhánh chính ko bị ảnh hưởng. Chỉ khi nào bạn commit những gì đã làm và push nó lên git server thì nó mới được vào nhánh chính. Và người khác khi lấy nội dung trên git về sẽ có nội dung mà bạn đã đưa lên.  
  
Hơi loằng ngoằng 1 chút, các bạn ko hiểu chỗ nào mail cho mình nhé! Bạn có thể thấy trên git server lúc này đã có file demo.txt

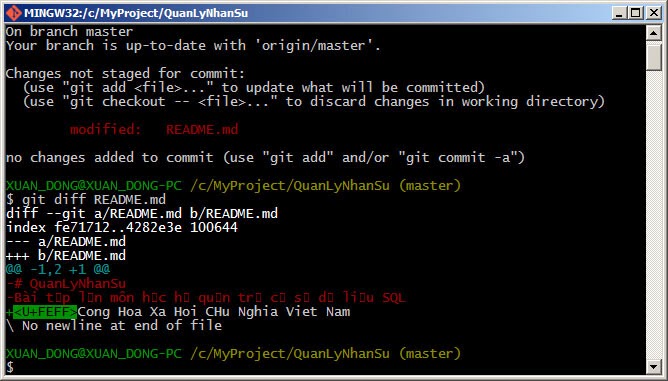
[](http://1.bp.blogspot.com/-EhfycH8zMik/VLaO-AI_HvI/AAAAAAAAAks/mmJ47i9JJS4/s1600/Untitled-1.jpg)

**3 - Sửa file đã có.**

Không phải thao tác nhiều như khi thêm file. Giả sử bạn sửa nội dung file "README.md" với nội dung file đó là "Cong hoa xa hoi chu nghia viet nam". Lưu file lại và sử dụng "git status" để xem sự thay đổi:

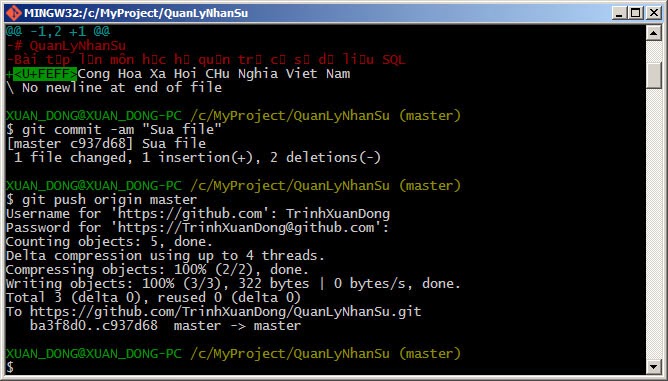
[](http://1.bp.blogspot.com/-KDY5OpD4E-w/VLaSRJb4wQI/AAAAAAAAAlA/IaMjvrQi4Jw/s1600/Untitled-2.jpg)

Ok, git báo "modified: README.md" => File này đã thay đổi. Tuy nhiên nếu bạn có nhu cầu xem lại xem mình đã thay đổi file đó như thế nào. Hãy sử dụng câu lệnh "git diff <tên file>", ví dụ "git diff README.md". Kết quả như sau

[](http://4.bp.blogspot.com/-3wwJebo-H9I/VLaSREKNEwI/AAAAAAAAAk8/KPg509ZWP-4/s1600/Untitled-3.jpg)

Git sẽ hiển thị ra một loạt các dòng nhiều màu khác nhau. Màu đỏ là ký hiệu bạn đã xóa dòng, còn màu trắng là bạn đã thêm dòng mới. Khi file đó dài và bạn sửa nhiều, sử dụng phím **lên**hoặc **xuống**để cuộn màn hình sao cho có thể quan sát đươc hết. Kết thúc ấn phím '**q**'. Việc xem diff file thực sự là quan trọng và hữu ích khi làm phần mềm. Vì khi bạn sửa 1 file nào đó, bạn lỡ sửa hàm abc nhưng bạn muốn lấy lại nguyên mẫu hàm abc này. Sử dụng **git checkout** như mục 4 thì ko ổn lắm vì sẽ làm mất toàn bộ những gì bạn đã sửa. Thay vì đó, hãy chịu khó xem**git diff** và khôi phục lại hàm abc bạn mong muốn.

Tiếp tục nhé, **git diff**ở trên chỉ là phụ lục cho bạn thôi. Sau khi bạn biết được có những file nào bị sửa đổi bằng câu lệnh **git status** ở trên, lần lượt thực hiện **git commit** **-am**và **git push origin master**như khi thêm file mới.

[](http://4.bp.blogspot.com/-53aswmP4LZg/VLaSQ8oMbxI/AAAAAAAAAk4/3qPuYsa0aAo/s1600/Untitled-4.jpg)

Kiểm tra lại trên git server xem sao nhé!

**4 - Hủy bỏ thay đổi đối với 1 hoặc nhiều file bạn đang làm việc.**

Khi bạn làm việc, nhiều khi bạn đang sửa nội dung một file nào đó, và bạn nhận ra đã có ai đó trong team của bạn đang sửa file này rồi, hoặc nội dung bạn sửa không đúng, và bạn muốn làm thế nào để file này hoàn nguyên về trạng thái ban đầu, như lúc chưa chỉnh sửa. Hãy sử dụng câu lệnh sau: "**git status**" để biết bạn đã thay đổi file nào, "**git checkout <tên file>** " để hoàn nguyên lại file đó. Trong trường hợp bạn muốn hoàn nguyên lại tất cả các file bạn đang sửa. Sử dụng lệnh "**git checkout .**" Ký tự "**.**" đại diện cho "tất cả các file". Câu lệnh này rất dễ sử dụng nên mình không demo!

**5 - Lấy dữ liệu mới nhất trên server về local.**

Trước khi lấy dữ liệu mới nhất trên server git, bạn phải đảm bảo project trên máy bạn thỏa mãn điều kiện đó là ko có file nào đang sửa nội dung hoặc khi bạn thêm 1 file mới mà bạn chưa commit.

Ví dụ như bạn đang sửa 1 file, nếu bạn ko commit file đó thì bạn ko thể lấy code mới nhất về được. Hoặc bạn có file mới, bạn dùng câu lệnh "**git add <file>**" nhưng lại ko commit file đó => Cũng ko thể lấy code trên server về.

Hãy sử dụng cách hủy bỏ thay đổi được nêu ra ở mục 4 bên trên để hoàn nguyên tất cả các file, hoặc add (nếu có file mới), commit thay đổi đó lên server như chỉ dẫn ở các mục trên. Nếu mọi thứ đã ok rồi thì dùng lệnh sau để lấy code mới nhất về: "**git pull --rebase origin master**". Ok, bây giờ dữ liệu trên máy local của bạn đã giống hệt như trên git server rồi đó.

**6 - Chú ý**

Bất kể khi bạn thêm 1 file mới hoặc sửa 1 file, trước khi push nó lên server hãy pull toàn bộ dữ liệu trên server về máy tính của mình đã. Why? Điều này đảm bảo rằng dữ liệu mới nhất trên server và dữ liệu bạn vừa thêm/sửa sẽ được đồng bộ với nhau trên máy bạn. Nó đồng nghĩa với việc nếu có bất kỳ lỗi nào đó xảy ra thì nó chỉ xảy ra trên máy bạn. Tránh bị lỗi sang máy của các thành viên trong nhóm. Hãy đảm bảo rằng dữ liệu trên máy mình không bị lỗi gì rồi hãy push lên server.

Nhìn chung quy trình khi bạn thêm  file mới sẽ như sau:

* git status
* git add <file>
* git commit -am "<Noi dung commit>"
* git pull --rebase origin master
* Kiểm tra lại dữ liệu xem có sai sót gì ko
* git push origin master

Còn khi sửa file thì quy trình như sau:

* git status
* git commit -am "<Noi dung commit>"
* git pull --rebase origin master
* Kiểm tra lại dữ liệu xem có sai sót gì ko
* git push origin master

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Vậy là mình đã kết thúc 4 phần hướng dẫn sử dụng github. Mong rằng các bạn có được nhiều kiến thức hơn. Good luck.

- See more at: http://csharpdepth.blogspot.com/2015/01/huong-dan-su-dung-github-part-3\_10.html#sthash.PapmybX6.dpuf