



ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Nhóm chuyên môn Nhập môn Công nghệ phần mềm

NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

Hướng dẫn bài tập:

Quản lý phiên bản với hệ thống Git 



1. Hệ thống quản lý phiên bản Git

2. Các lệnh thao tác trên Git

Khung hình giảng viên
xuất hiện khi ghi hình bài giảng
*(Thầy/Cô không chèn
văn bản/hình ảnh vào đây)*

MỤC TIÊU



Sau bài học này, người học có thể:

1. Hiểu về **cấu trúc và cơ chế làm việc** trên hệ thống quản lý phiên bản Git
2. Nắm được **các lệnh thao tác** trên hệ thống Git

Khung hình giảng viên
xuất hiện khi ghi hình bài giảng
(Thầy/Cô không chèn
văn bản/hình ảnh vào đây)

4.44''

1. Hệ thống quản lý phiên bản Git

1.1. Giới thiệu về Git

1.2. Cài đặt Git và các thiết lập cơ bản

2. Các lệnh thao tác trên Git

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT

1.1. Giới thiệu về Git

- Git^[1] là một hệ thống quản lý phiên bản phân tán
- Là hệ thống quản lý phiên bản phổ biến nhất
- Dễ sử dụng, an toàn và nhanh chóng

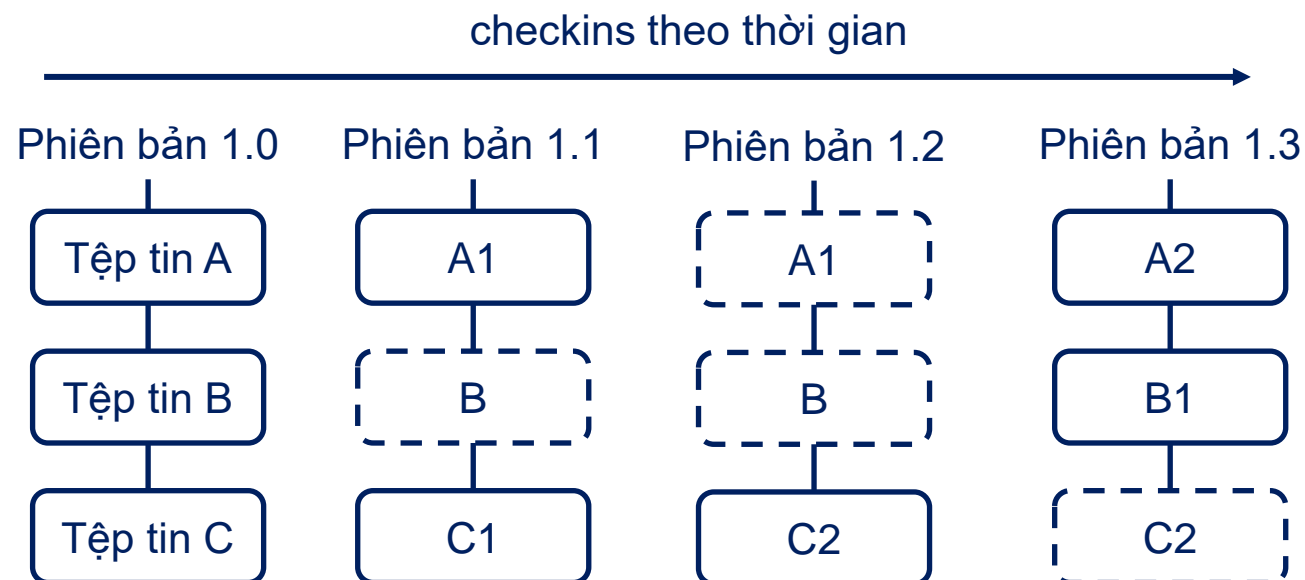


Hình 1.1: Logo của Git [1]

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT

1.1. Giới thiệu về Git

- Cơ chế làm việc: Git lưu trữ dữ liệu như một **luồng các ảnh chụp nhanh (stream of snapshots)** trạng thái của dự án theo thời gian
- Nếu tệp không thay đổi, Git chỉ lưu một liên kết đến tệp đã được lưu trữ trước đó

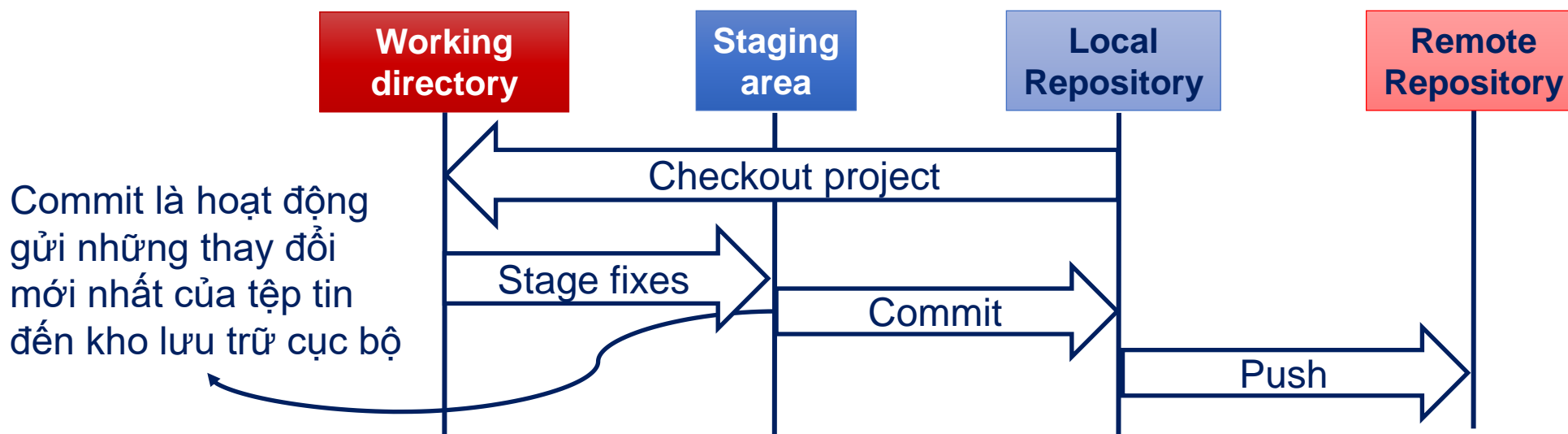


Hình 1.2: Git lưu trữ dữ liệu dưới dạng ảnh chụp nhanh của dự án theo thời gian

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT

1.1. Giới thiệu về Git

- Các phần chính của một dự án Git (Git project):
 - Thư mục làm việc (Working directory / Working tree)
 - Vùng tổ chức hay khu vực trung gian (Staging area / Index)
 - Kho lưu trữ (Repository / Git directory): cục bộ (local) và từ xa (remote)



Hình 1.3: Các phần chính của một dự án Git

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT



1.1. Giới thiệu về Git

- Trạng thái của các tệp tin trong dự án Git:
 - Chưa được theo dõi (Untracked): tệp tin chưa được theo dõi phiên bản
 - Đã theo dõi (Tracked): tệp tin được đánh dấu để Git quản lý phiên bản. Tại trạng thái này, tệp tin có thể nhận thêm 3 trạng thái cụ thể khác:
 - Đã chỉnh sửa (Modified): tệp tin được thay đổi nhưng chưa được commit vào kho lưu trữ, cần chuyển vào trạng thái Staged
 - Sẵn sàng để commit (Staged): đánh dấu tệp tin đã được chỉnh sửa trong phiên bản hiện tại để đưa vào snapshot lần commit tiếp theo
 - Đã hoàn thành commit (Committed): dữ liệu (các thay đổi) đã được lưu trữ an toàn trong kho lưu trữ cục bộ

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT



1.2. Cài đặt Git và các thiết lập cơ bản

- Bài tập: Cài đặt Git vào máy tính và thực hiện các thiết lập cơ bản
- Hướng dẫn:
 - Trên hệ điều hành Windows và macOS, tải về bộ cài đặt Git và thực hiện cài đặt: <https://git-scm.com/downloads>
 - Trên hệ điều hành Linux (Ubuntu/Debian), có thể sử dụng lệnh

```
$ sudo apt-get install git
```
 - Trên hệ điều hành Linux (CentOS/Fedora/RHEL), có thể sử dụng lệnh

```
$ yum install git
```

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT

1.2. Cài đặt Git và các thiết lập cơ bản

- Thao tác với Git được thực hiện thông qua giao diện dòng lệnh hoặc sử dụng các công cụ có giao diện người dùng đồ họa (GUI)
- Sau khi cài đặt Git vào hệ điều hành Windows, mở ứng dụng **Git Bash** để bắt đầu sử dụng các lệnh của Git

\$ git --version ←

Lệnh hiển thị phiên bản Git được cài đặt trên máy



```
Admin@DESKTOP-JTD8FOD MINGW64 ~  
$ git --version  
git version 2.38.1.windows.1  
Admin@DESKTOP-JTD8FOD MINGW64 ~  
$
```

Hình 1.4: Giao diện dòng lệnh của Git Bash trên hệ điều hành Windows

1. HỆ THỐNG QUẢN LÝ PHIÊN BẢN GIT

1.2. Cài đặt Git và các thiết lập cơ bản

- Cấu hình thông tin cá nhân: khai báo tên và địa chỉ email vào trong **file cấu hình của Git** trên máy (tệp tin **.gitconfig**)

```
$ git config --global user.name "Ten"
```

```
$ git config --global user.email "Email"
```

- Xem lại danh sách các thiết lập của Git trên máy hiện tại

```
$ git config --list
```

- Xem nội dung tệp tin cấu hình

```
$ cat ~/.gitconfig
```

1. Hệ thống quản lý phiên bản Git

2. Các lệnh thao tác trên Git

2.1. Tạo kho lưu trữ (repository)

2.2. Một số lệnh cơ bản

2.3. Làm việc với nhánh (branch)

2.4. Vấn đề xung đột

2. CÁC LỆNH THAO TÁC TRÊN GIT



2.1. Tạo kho lưu trữ (repository)

- Bài tập: Tạo kho lưu trữ cục bộ (local repository) tại thư mục mã nguồn của dự án
- Hướng dẫn:
 - **Bước 1:** Truy cập vào thư mục mã nguồn, sử dụng lệnh **cd**
 - **Bước 2:** Khởi tạo kho lưu trữ tại thư mục hiện tại, sử dụng lệnh **git init**
 - (Tạo thư mục mới và khởi tạo kho lưu trữ tại thư mục đó, sử dụng lệnh: **git init tên_thư_mục**)
- Ví dụ:

```
MINGW64/d/Sample projects/SimpleMVCApp
Admin@DESKTOP-JTD8F0D MINGW64 /d/Sample projects
$ cd SimpleMVCApp/

Admin@DESKTOP-JTD8F0D MINGW64 /d/Sample projects/SimpleMVCApp (master)
$ git init
Reinitialized existing Git repository in D:/Sample projects/SimpleMVCApp/.git/

Admin@DESKTOP-JTD8F0D MINGW64 /d/Sample projects/SimpleMVCApp (master)
$
```

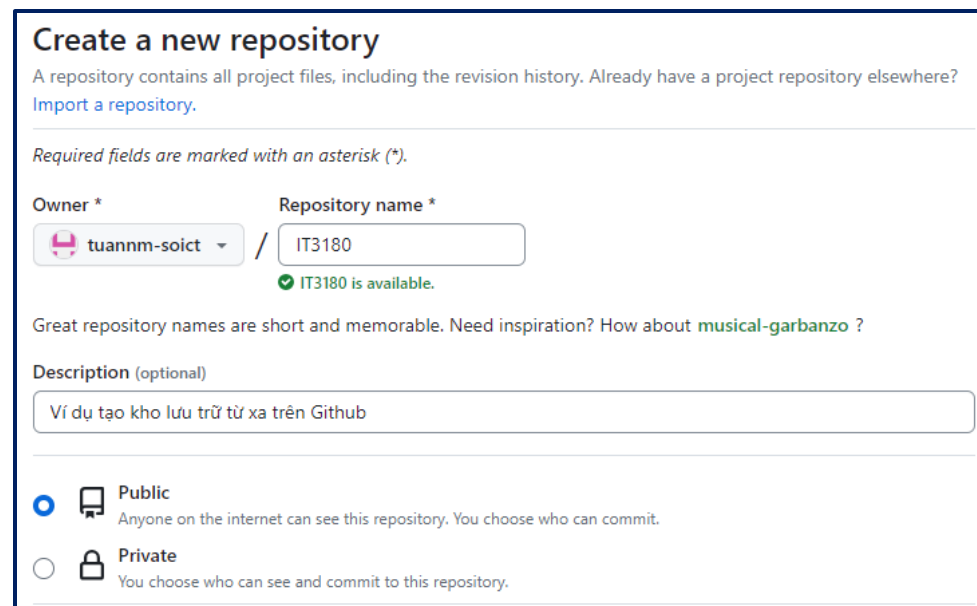
Hình 2.1: Khởi tạo kho lưu trữ cục bộ

2. CÁC LỆNH THAO TÁC TRÊN GIT

2.1. Tạo kho lưu trữ (repository)

- Bài tập: Tạo kho lưu trữ từ xa (remote repository) trên Github^[2] và tạo bản sao về máy cá nhân
- Hướng dẫn:
 - **Bước 1:** Đăng ký tài khoản miễn phí và đăng nhập vào <https://github.com/>
 - **Bước 2:** Tạo mới kho lưu trữ
 - Sau khi tạo xong, địa chỉ kho lưu trữ có dạng:

[https://github.com/\\$user-name/\\$repository](https://github.com/$user-name/$repository)




Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner * / Repository name *


 tuannm-soict / IT3180


✔ IT3180 is available.

Great repository names are short and memorable. Need inspiration? How about [musical-garbanzo](#) ?

Description (optional)

Ví dụ tạo kho lưu trữ từ xa trên Github

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Hình 2.2: Tạo mới kho lưu trữ từ xa trên Github

2. CÁC LỆNH THAO TÁC TRÊN GIT

2.1. Tạo kho lưu trữ (repository)

- Hướng dẫn:

- **Bước 3:** Tạo bản sao của kho chứa trên Github về máy, sử dụng lệnh:

\$ git clone địa_chỉ_kho_lưu_trữ

- Ví dụ: **\$ git clone https://github.com/tuannm-soict/IT3180**

```
MINGW64:/d/Sample projects/it3180/IT3180
Admin@DESKTOP-JTD8FOD MINGW64 /d/Sample projects/it3180
$ git clone https://github.com/tuannm-soict/IT3180
Cloning into 'IT3180'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

Hình 2.3: Tạo bản sao từ kho lưu trữ trên Github

2. CÁC LỆNH THAO TÁC TRÊN GIT



2.2. Một số lệnh cơ bản

- Lệnh thêm các tệp tin có trong kho chứa để quản lý phiên bản (thiết lập trạng thái tracked)

`$ git add tên_file`

- Lệnh kiểm tra trạng thái các tệp tin được đánh dấu quản lý

`$ git status`

- Lệnh ghi lại việc thêm / thay đổi của tệp tin

`$ git commit -m "Nội dung lời nhắn"`

- Với các kho chứa từ xa như Github, cần đẩy các tệp tin đã được commit

`$ git push origin [branch]`

- Cập nhật kho lưu trữ: `$ git pull`

2. CÁC LỆNH THAO TÁC TRÊN GIT



2.2. Một số lệnh cơ bản

- Xem lịch sử của các lần commit trước đó

\$ git log

- Sử dụng tham số -p để hiển thị chi tiết của mỗi lần commit

\$ git log -p

- Một số tùy chọn xem log:
 - --since, --after: Xem các lần commit kể từ ngày nhất định
 - --until: Xem các lần commit trước từ ngày nhất định
 - --author: Xem các lần commit của một người cụ thể
 - --grep: Lọc các chuỗi trong log

2. CÁC LỆNH THAO TÁC TRÊN GIT



2.3. Làm việc với nhánh (branch)

- Nhánh chính:
 - **Nhánh master**: lưu trữ lịch sử phát hành chính thức (production)
 - **Nhánh develop**: nhánh tích hợp cho các tính năng (develop)
 - **Nhánh staging**: được tách ra từ develop để đảm bảo chất lượng kiểm tra
- Thao tác với nhánh:
 - Lệnh tạo một nhánh: **\$ git branch [branch]**
 - Lệnh xóa một nhánh: **\$ git branch -d [branch]**
 - Lệnh chuyển nhánh khác:
 - \$ git checkout [branch]**
 - \$ git checkout -b [new_branch]**

2. CÁC LỆNH THAO TÁC TRÊN GIT



2.3. Làm việc với nhánh (branch)

- HEAD - con trỏ cho biết vị trí hiện tại đang nằm ở đâu trong working tree
- Gộp dữ liệu từ một branch:
 - Hợp nhất các nhánh (Merge): chuyển dữ liệu từ một branch nào đó về branch đang trỏ đến
 - `$ git merge [branch]`
 - Tích hợp 2 nhánh, tạo ra 1 merge commit và thêm vào nhánh đích
 - Cây phân nhánh có thể rối, khó nhìn
 - Tìm kiếm và sửa lỗi dễ dàng

2. CÁC LỆNH THAO TÁC TRÊN GIT



2.4. Vấn đề xung đột

- Xung đột xảy ra khi hai hay nhiều người thay đổi cùng một tệp
- Phát hiện khi xử lý merge request
- Git sẽ đánh dấu file bị xung đột và tạm dừng quá trình hợp nhất
- Sau đó, trách nhiệm của các nhà phát triển là giải quyết xung đột dựa trên những gì Git đã đánh dấu

```
Accept Current Change | Accept Incoming Change | Accept Both Changes |
<<<<<< HEAD (Current Change)
    <div>Github</div>
=====
    <div>Gitlab</div>
>>>>>> develop (Incoming Change)
```

Hình 2.4: Minh họa xung đột khi hợp nhất

1. Bài học đã cung cấp cho người học **giới thiệu tổng quan** về hệ thống quản lý phiên bản phân tán Git và **các lệnh cơ bản** trên hệ thống này
2. Tiếp sau bài này, **người học có thể tự thực hành và tìm hiểu thêm** về các lệnh chuyên sâu trên Git

NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

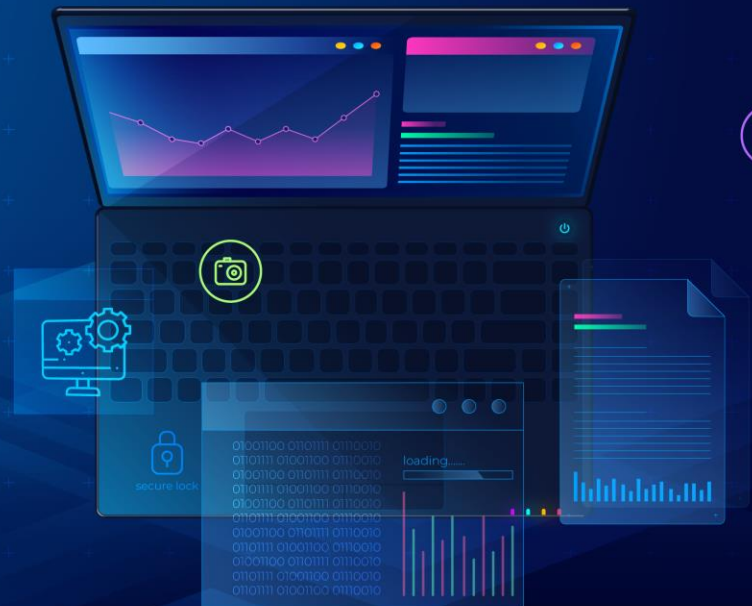
Quản lý phiên bản với hệ thống Git

Biên soạn:

ThS. Nguyễn Mạnh Tuấn

Trình bày:

ThS. Nguyễn Mạnh Tuấn



NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

Bài học tiếp theo:

Hướng dẫn, giới thiệu một số tài liệu trong quy trình phát triển phần mềm

Tài liệu tham khảo:

- [1] R. Pressman, Software Engineering: A Practitioner's Approach. 8th Ed., McGraw-Hill, 2016 và bộ slide đi kèm.
- [2] I. Sommerville, Software Engineering. 10th Ed., AddisonWesley, 2017.
- [3] Pankaj Jalote, An Integrated Approach to Software Engineering, 3rd Ed., Springer.
- [4] Shari Lawrence Pleege, Joanne M. Atlee, Software Engineering theory and practice. 4th Ed., Pearson, 2009

Tư liệu:

- [1] git. Available at: <https://git-scm.com/> (Accessed: 27 September 2023).
- [2] Github. Available at: <https://github.com/> (Accessed: 27 September 2023).