**Git Flow & Azure Repos**

# GIT FLLOW

## Git Flow là gì?

Git Flow là luồng đi quản lý source code nhằm cải thiện quá trình làm việc thông qua mô hình phân nhánh và hoạt động như một kho lưu trữ trung tâm cho một dự án. Dưới đây là sơ đồ hoạt động của Git Flow.

Chart, diagram

Description automatically generated with medium confidence

Khởi tạo nhánh chính (Master), sau đó khởi tạo nhánh Develop (nhánh phục vụ cho việc tạo sản phẩm), từ nhánh Develop tạo nhánh con Feature (làm từng chức năng), sau khi hoàn thành các chức năng thì code sẽ được đẩy lên Pull Request để được review code, sau khi review hoàn thành thì code được leader merge code vào nhánh Develop, sau đó code được đưa lên môi trường test, sau khi test thành công thì leader sẽ đẩy code lên nhánh Release (gửi khách hàng comfirm), sau đó đẩy code sang nhánh Master (từ đây, code được triển khai đưa lên server product đến người dùng). Sau khi code được đẩy lên Master phát sinh ra lỗi thì sẽ phải tạo một nhánh Hotfix để sửa code, sau khi hoàn thành fix code thì đẩy code lên lại Master và đồng thời chuyển code về Develop.

## Các branch

### Master

Là branch tồn tại xuyên suốt quá vòng đời của phần mềm được tạo mặc định trong Git khi ta tạo repository.

Đây là branch chính của cả dự án chỉ có thể merge vào từ branch release chứ không được code trực tiếp trên branch này. Sản phẩm được đưa lên production sẽ được lấy ở đây nên code được merge vào master phải được quản lý và xử lý cẩn thận nhất có thể.

### Develop

Là nơi phát triển chính và branch luôn tồn tại song song với master.

Branch develop là branch trung tâm cho việc phát triển. Mỗi khi có tính năng mới cần xây dựng thì branch feature sẽ được tách từ branch develop để phát triển. Mỗi khi chuẩn bị release phiên bản mới thì branch release sẽ được tách từ branch develop để test

### Feature

Là nhánh được tách từ develop nhằm mục đích xây dựng các tính năng riêng mà không phụ thuộc vào nhau.

Các branch feature được sử dụng để phát triển các tính năng mới cho bản release sắp tới hoặc trong tương lai xa. Branch feature sẽ tồn tại miễn là tính năng này đang được phát triển, nhưng cuối cùng sẽ được hợp nhất trở lại Branch Develop. Ngoài các tính năng chính thì còn có thể tách thành cách tính năng nhỏ hơn để thuận tiện cho việc phát triển tùy vào cấu trúc cũng như độ phức tạp của dự án.

### Release

Là nhánh tách từ develop để kiểm tra và fix bug chuẩn bị cho việc ra mắt sản phẩm.

Branch release là branch dùng để release sản phẩm như đúng tên gọi của nó. Khi khách hàng cần release một số tính năng thì một branch sẽ được tách từ branch develop ra với tên theo cấu trúc "release/v.1.0.1" . Sau đó sẽ test và fix bug trên branch release, khi xong sẽ merge vào branch master để đẩy sản phẩm lên và merge vào branch develop để tránh gặp lại các bug đã được test và fix.

### Hotfix

Là nhánh tách từ master để fix gấp những bug còn tồn đọng mà trên release chưa xử lý hết.

Có 2 trường hợp thường xảy ra:

- Có những bug mà trong quá trình sử dụng thực tế người dùng phát hiện ra mà QA không test được.

- Đôi lúc vì một số lý do khách hàng muốn ra mắt ngay tính năng mới đang phát triển trên develop. Và tất nhiên khi đẩy trực tiếp như thế sẽ xuất hiện một lỗi nghiêm trọng trong phiên bản release.

-> Vì code trên master phải là code release được nên nếu có bug nó phải được ưu tiên sửa chữa ngay. Từ master branch hotfix sẽ được tạo và xử lý khi xong sẽ lại được merge vào master và develop.

## Một số lệnh

### Tạo một kho chứa Git

**Syntax:** $ git init

Lệnh này sẽ tạo một thư mục mới có tên .git, thư mục này chứa tất cả các tập tin cần thiết cho kho chứa.

### Sao chép một kho chứa đã tồn tại

**Syntax:** $ git clone <https://github.com/user/repository.git>

Câu lệnh trên sẽ tạo một thư mục mới có tên giống trên của Repo.

### Nhánh trong git

* Câu lệnh Git dùng để kiểm tra branch hiện tại: *$ git branch*
* Để tạo mới một branch: *$ git branch <name\_branch>*
* Để chuyển và tạo mới*: $ git branch -b <name\_branch>*

### Chuyển nhánh

Để checkout một nhánh: *$ git checkout <name\_branch>*

### Cập nhật thay đổi

Sau khi thay đổi source code: thêm mới, sửa, xoá files,… Bạn cần phải cập nhật lên Staging Area. Để cập nhật hết các files: *$ git add*

Sau lệnh add, bạn cần sử dụng câu lệnh Commit để đây thông tin thay đổi lên Local Respository*: $ git commit -m "Message"*

### Cập nhật lên server

Sau câu lệnh Commit, thông tin mới chỉ được cập nhật lên Local Repository. Nếu muốn cập nhật lên server thì phải sử dụng câu lệnh push: *$ git push origin <name\_branch>*

Ngoài ra, nếu chưa tồn tại remote trên server thì bạn cần phải add mới một remote trước rồi mới push:

*$ git remote add origin <remote\_url>*

*$ git push origin <name\_branch>*

### Gộp nhánh

Sau một thời gian cập nhật các file và push lên git trên branch mới, bây giờ mình cần ghép (merge) code lại vào nhánh gốc (master). Trước tiên, cần phải checkout ra khỏi branch hiện tại cần gộp để vào branch master, sau đó thì dùng lệnh merge để ghép branch mới vào master:

*$ git checkout master*

*$ git merge <new\_branch>*

### Xem lại lịch sử commit

Biết về người commit, ngày giờ, message của những lần commit đó: *$ git log*

### Xem thay đổi trước khi push

*$ git diff*

Lệnh này giúp biết những gì đã được thay đổi giữa nhánh hiện tại và nhánh trước nó.

### Pull từ remote repository

*$ git pull origin master*

Lệnh này sẽ gộp những thay đổi mới kéo về từ máy chủ từ xa với nhánh hiện tại trên máy local.

### Phân biệt git pull & git fetch

Khi thực hiện clone một dự án từ remote repository, thì các dữ liệu đó chỉ là một “snapshot” tại thời điểm bạn clone dự án mà thôi. Do vậy, khi có các thay đổi trên remote repository thì chúng cần được tải về trong local repository Lúc này chúng ta có thể sử dụng\*\* git fetch\*\* hoặc \*\*git pull\*\* để thực hiện việc này.

|  |  |
| --- | --- |
| Git fetch | Git pull |
| git fetch <remote\_origin> | **git <remote\_origin> <branch\_name>** |
| Git sẽ tải về dữ liệu của tất cả các branch của repository trên remote server nằm tại địa chỉ quy định bởi <remote\_origin> và cập nhật dữ liệu này với dữ liệu của cách branch phía dưới máy local. Tuy nhiên git fetch không cập nhật dữ liệu của working directory. Điều này có nghĩa là nếu như có bất cứ thay đổi (commit) nào trên remote server thì chúng cũng không ảnh hưởng tới các tập tin, thư mục | **Git sẽ thực hiện việc fetch dữ liệu của remote repository tại nhánh <branch\_name> từ server nằm tại địa chỉ quy định bởi <remote\_origin> và áp dụng (merge) các thay đổi này vào thư mục và tập tin ở working directory.**  **Git pull luôn cố gắng tự động trộn các thay đổi (auto merge) vì thế việc này có thể dẫn đến conflict.** |

### Phân biệt git reset & git revert

Trong quá trình phát triền phần mềm, thì chúng ta thường không thể tránh khỏi việc sử dụng phải rollback một số commit bị lỗi. Trong trường hợp này, chúng ta sẽ có 2 phương pháp chính để giải quyết, đó là: git reset và git revert.

|  |  |
| --- | --- |
| Git reset | Git revert |
| Git reset được dùng để quay về một điểm commit nào đó, đồng thời xóa lịch sử của các commit trước nó. | Git revert không làm mất các commit , thay vào đó sẽ tạo thêm commit mới, có nội dung giống hệt với commit bạn muốn quay trở về |

### Lệnh git rebase

Lệnh git rebase cũng gộp các commit từ nhánh này vào nhánh khác, bằng cách xây dựng lại các commit base kế thừa từ nhánh khác và viết lại lịch sử commit sau các commit cơ sở mới.

git checkout <branch\_name>

git rebase <branch\_name\_base>

### Lệnh git cherry-pick

git cherry-pick tạo ra một bản copy của một commit từ một nhánh khác và apply vào nhánh hiện tại với cùng message và thay đổi từ commit đó.

# Azure repos

## Tổng quan về Azure

Microsoft Azure được định nghĩa là một nền tảng điện toán đám mây và được phát hành vào năm 2020 bởi chuyên gia công nghệ Microsoft. Azure mang lại giải pháp Cloud tích hợp toàn diện để sử dụng cho việc xây dựng, triển khai cũng như quản lý cho các ứng dụng thông qua mạng lưới trung tâm toàn cầu của Microsoft. Nó là nền tảng có phạm vi sử dụng các dịch vụ trực tuyến hoặc ứng dụng vô cùng rộng lớn.

Azure DevOps là một trong những kho lưu trữ riêng miễn phí và không có giới hạn. Nó cho phép bạn theo dõi lỗi, thông tin phản hồi cũng như hạng mục công việc,... Nhà phát triển có thể làm việc dựa trên đám mây nhờ vào sử dụng Azure DevOps Service; hoặc làm tại on-premise bằng việc sử dụng Azure DevOps Server.

Một số bộ tính năng tích hợp của Azure DevOps:

* Azure Boards: Nó cung cấp bộ công cụ Agile để lập kế hoạch và theo dõi công việc bằng Kanban và Scrum.
* Azure Repos: cung cấp cho kho kiểm soát phần mềm Git hoặc Team Foundation Version Control (TFVC) thực hiện kiểm soát nguồn mã.
* Azure Artifacts: Cho phép bạn chia sẻ các gói Maven, npm và NuGet từ những nguồn công khai và cá nhân.
* Azure Pipelines: Nó cung cấp các dịch vụ phát hành và xây dựng để tích hợp và chuyển giao liên tục (CI/CD).
* Azure Test Plans: Cung cấp các công cụ cho việc kiểm tra ứng dụng

## Thành phần trong Azure Repos

### Files

Chứa toàn bộ thư mục của dự án

Bấm nút clone để copy đường dẫn, sau đó viết lệnh: $ git clone <paste\_URL>

### Commits

Lưu lại toàn bộ lịch sử hoạt động của sự thay đổi trong thư mục làm việc. Mỗi lần commit nó sẽ lưu lại lịch sử chỉnh sửa của code kèm theo tên và địa chỉ email của người commit.

Tips commit:

* Những thay đổi mang ý nghĩa khác nhau chẳng hạn như thêm chức năng hay sửa lỗi thì hãy cố gắng chia ra rồi commit. Để sau này khi xem lịch sử và tìm kiếm một nội dung thay đổi định sẵn sẽ dễ dàng hơn.
* Vì phần giải thích đó sẽ trở thành thông tin quan trọng khi bản thân xem lại lịch sử sau đó hay trường hợp tìm kiếm nội dung thay đổi mà người khác commit, nên hãy chú ý để viết giải thích sao cho dễ dàng hiểu được nội dung thay đổi. Viết giải thích theo hình thức tiêu chuẩn trên Git như sau:
* Dòng thứ 1: Tóm tắt nội dung thay đổi trong commit
* Dòng thứ 2: Dòng trống
* Dòng thứ 3 trở đi: Lý do đã thay đổi

### Pushes

Text

Description automatically generated

### Branchs

Thêm mới branch bằng UI(New branch) trên browser

VS2019: Git 🡪 New branch

Git cmd: git branch <new branch name>

### Pull Requests

Pull request được tạo ra để đưa những file source code của lên 1 host chung nơi mọi người có quyền truy cập sẽ truy cập vào và cùng review, để lại comment trên những file source code đó.

Thêm mới PR bằng UI trên browser