



 **A2 HOSTING**

**BLAZING FAST  
CLOUD HOST**  
DESIGN YOUR **PERFECT CLOUD**

- Quick Start Cloud
- 300% Faster On SSDs
- Free High Availability

**CLOUD VPS  
\$15.00/mo**  
**Learn More**

# [Git] Branch – Kỹ thuật phân nhánh



Cập nhật lúc 17/09/2016



No Comments



Công cụ , Web Development

Bài này thuộc phần 9 của 11 phần trong serie [Git cơ bản](#)

5 CHIA SẺ



Facebook



Twitter

2



Google+

0



Pocket

3

Trong khi làm việc với Git, bạn đã quá quen thuộc với việc chỉnh sửa mã nguồn, sau đó là commit mỗi khi chỉnh sửa xong và push lên remote repository nếu cần thiết. Nhưng bây giờ mình có một ví dụ đặt ra là mình muốn **tạo một phiên bản thử nghiệm với mã nguồn đang làm việc trong working tree hiện tại mà không gây ảnh hưởng đến các code hiện tại**. Vậy thì làm cách nào? Không lẽ clone một repository từ chính cái working tree hiện tại rồi sửa đổi hay sao? Như thế rất là mất công, mà lại không tối ưu và không thể đồng bộ hóa hoặc rất khó

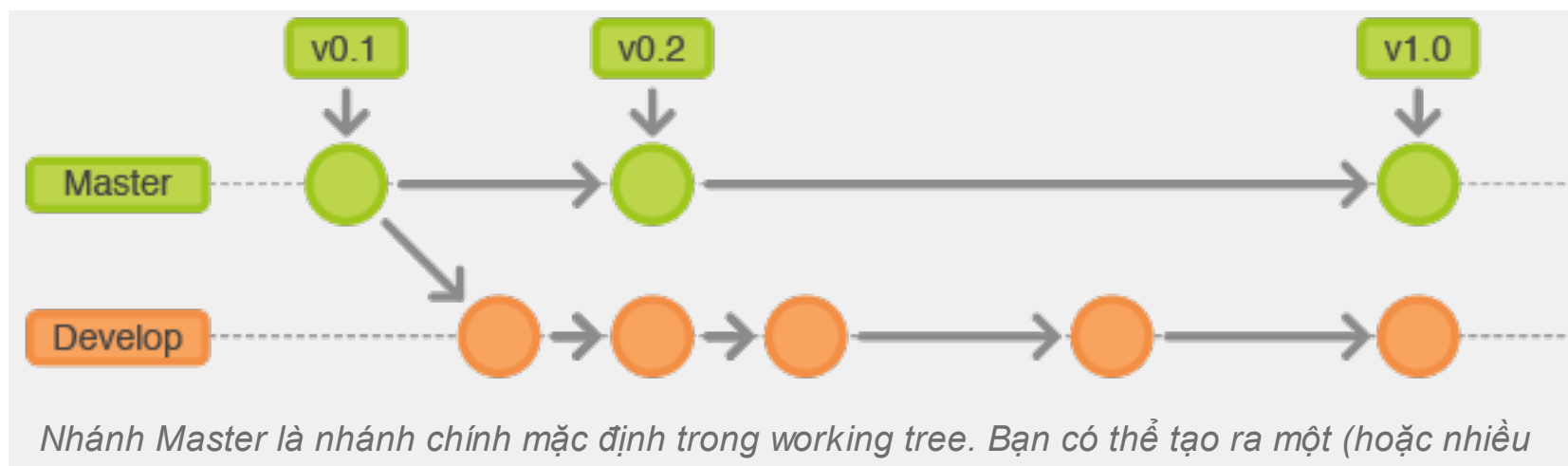
khăn để đồng bộ hóa.

Mà trong Git, chúng ta sẽ sử dụng một giải pháp khác tuyệt vời hơn, dễ dàng hơn gọi là **phân nhánh** (*branching*), mà cụ thể là phân nhánh cái gì? Đó là phân nhánh trong working tree hiện tại bạn đang làm việc đấy, và mỗi nhánh chúng ta sẽ gọi nó là một **branch**.

## Branch trong Git là gì?

Khi bắt đầu khởi tạo một repository hoặc clone một repository, bạn sẽ có một nhánh (branch) chính tên là `master` (bạn có thể hiểu master là một cái thân cây). Đây là branch mà sẽ chứa toàn bộ các mã nguồn chính trong repository.

Đó là lý do tại sao, ở các phần trước khi chúng ta push hoặc pull hay làm một số việc khác thì lại có tham số `master` trong câu lệnh, đó nghĩa là chúng ta đang thực hiện thao tác trên branch `master`.



nhánh mới) với tên là *Develop* chẳng hạn.

Bây giờ nếu bạn muốn tạo một sự thay đổi nào đó mà có thể trực tiếp sử dụng trên working tree hoặc commit, push lên repository mà không ảnh hưởng đến branch master thì sẽ cần tạo ra một branch mới với tên là `develop` chẳng hạn. Và từ đó mỗi khi bạn thực hiện lệnh checkout vào branch nào đó thì toàn bộ mã nguồn trên working tree của bạn sẽ được đổi sang môi trường dành cho branch đang checkout.

Giờ mình lấy một ví dụ đơn giản, bên branch `master` bạn tạo một tập tin *master.html* rồi commit lên, lúc này khi bạn qua cái branch `develop` (đã được tạo trước đó) thì cái tập tin *master.html* kia sẽ không có vì nó đã được commit bởi branch `master`, không liên quan gì tới `develop`. Tương tự, các thay đổi của bạn bên `develop` cũng sẽ không ảnh hưởng gì tới bên `master` cả. Điều này có một cái thú vị là nó tương tác trực tiếp trên máy tính của bạn, ví dụ tập tin *index.html* của `master` có nội dung khác và *index.html* bên `develop` thì khi bạn dùng lệnh checkout là nội dung trên máy tính nó tự đổi tương ứng, đó là lý do bạn có thể test nhiều phiên bản trên máy tính mà không cần đổi thư mục, chỉ cần checkout cái branch.

Giải thích thì hơi dài dòng, cứ vào làm ví dụ cụ thể cho dễ hiểu nhé.

## HEAD – con trỏ vị trí

Trước khi nói tiếp về branch thì mình xin nói qua một xíu về `HEAD`. Trong Git, từ khóa HEAD sẽ tượng trưng cho con trỏ chỉ cho bạn biết bạn đang nằm ở đâu. Bây giờ hãy xem lại bài hướng dẫn đọc log commit của mình, tìm

tới phần tối ưu log và cài vào. Sau đó gõ lệnh `git lg` thì bạn sẽ thấy từ khóa HEAD cho bạn biết bạn đang ở đâu trong working tree.

```
thachpham@ubuntu:~$ cd hoc-git
thachpham@ubuntu:~/hoc-git$ git lg
* 83e9976 - (HEAD, master) Add version1.txt by master (6 hours ago) <Thach Pham>
* d5a599e - (tag: v1.0-an, thach/version1) Commit for Annotated Tag (6 hours ago) <Thach Pham>
* 0519337 - (tag: v1.0) Added a new tag (7 hours ago) <Thach Pham>
* 435f64 - (thach/master) Added faq.html (8 hours ago) <Thach Pham>
* 6904d52 - (tag: v0.0) Initial commit (8 hours ago) <Thach Pham>
thachpham@ubuntu:~/hoc-git$
```

Như ảnh trên thì có nghĩa là mình đang ở branch `master`.

Ngoài ra còn một cách khác để kiểm tra đó là đọc tập tin `.git/HEAD`

```
$ cat .git/HEAD
ref: refs/heads/master
```

## Cách tạo một branch

Trước tiên bạn có thể xem toàn bộ các branch mà bạn đang có trong working tree bằng lệnh `git branch`. Sau đó nếu muốn tạo thêm branch, chỉ cần gõ lệnh `git branch tên_brand`. Ví dụ mình cần tạo branch `develop`.

```
$ git branch develop
```

Bây giờ bạn có thể gõ lại lệnh `git branch` một lần nữa để xem sẽ thấy brand tên develop xuất hiện.

## Checkout một branch

Checkout ở đây nghĩa là bạn truy cập kiểm tra mã nguồn trong branch đó để làm việc đấy. Để làm việc này, bạn sử dụng lệnh `git checkout tên_branch`.

```
$ git checkout develop  
Switched to branch 'develop'
```

Lúc này bạn đã đổi sang branch `develop` rồi, để kiểm tra chắc chắn bạn có thể gõ các lệnh kiểm tra HEAD ở trên.

Bây giờ bạn sẽ làm việc trong branch mới chuyển hay nói đúng hơn là bạn đang làm việc ở chỗ mà cái HEAD đang trỏ tới. Để chuyển về branch chính thì gõ `git checkout master`.

Bây giờ bạn thử tạo một tập tin nào đó, sau đó commit ở branch `develop` rồi chuyển về branch master sẽ thấy những gì bạn đã làm ở branch `develop` hoàn toàn vô nghĩa ở `master`. Dưới đây là ví dụ về việc ở branch `master` không có tập tin develop.html được tạo ra từ branch `develop`.

```
$ touch develop.html
$ git add .
$ git commit -m "Commit develop.html from develop branch"
[develop 8c68896] Commit develop.html from develop branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 develop.html
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'thach/master' by 3 commits.
 (use "git push" to publish your local commits)
$ ls
faq.html README.md tag.html version1.txt
```

Tương tự với việc sửa tập tin hay bất kỳ làm việc gì khác nó cũng chỉ áp dụng thay đổi ở branch bạn đang trở tới.

## Gộp dữ liệu từ một branch

Nếu mỗi branch nó nằm riêng như vậy thì bạn muốn sử dụng các thay đổi ở một branch nào đó cho master thì sao? À, chúng ta có thể sử dụng lệnh git merge để chuyển dữ liệu từ một branch nào đó về branch mà bạn đang trở đến. Lưu ý là ở branch cần chuyển về đã phải được commit. Ví dụ mình cần chuyển dữ liệu từ branch develop về master thì sẽ làm lần lượt các lệnh sau:

```
$ git checkout master
Already on 'master'
Your branch is ahead of 'thach/master' by 3 commits.
```

```
(use "git push" to publish your local commits)
$ git merge develop
Updating 83e9976..8c68896
Fast-forward
 develop.html | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 develop.html
$ ls
develop.html faq.html README.md tag.html version1.txt
```

## Xóa branch

Nếu bạn không cần dùng tới branch nào nữa thì có thể xóa với lệnh `git branch -d tên_branch`. Lưu ý là cái branch muốn xóa phải được gộp dữ liệu (merge) về master.

```
$ git branch -d develop
Deleted branch develop (was 8c68896).
```

Sau khi xóa xong, nó sẽ báo branch đó đã được móc vào commit với mã checksum nào (8c68896).

## Làm việc với remote branch

Quay lại một xíu với bài remote repository, bây giờ bạn hãy tạo thêm một remote mới từ địa chỉ <https://github.com/csswizardry/inuit.css> và đặt tên cho remote này là `inuit` vào working tree của bạn.

```
$ git remote add inuit https://github.com/csswizardry/inuit.css
$ git remote -v
inuit https://github.com/csswizardry/inuit.css (fetch)
inuit https://github.com/csswizardry/inuit.css (push)
thach https://github.com/thachphamblog/hoc-git.git (fetch)
thach https://github.com/thachphamblog/hoc-git.git (push)
```

Bây giờ bạn có thể xem toàn bộ branch của cái remote `inuit` mới thêm vào bằng lệnh `git remote show inuit`.

```
$ git remote show inuit
* remote inuit
Fetch URL: https://github.com/csswizardry/inuit.css
Push URL: https://github.com/csswizardry/inuit.css
HEAD branch: master
Remote branches:
feature/docs new (next fetch will store in remotes/inuit)
feature/grids new (next fetch will store in remotes/inuit)
fix/beautons new (next fetch will store in remotes/inuit)
fix/grids new (next fetch will store in remotes/inuit)
gh-pages new (next fetch will store in remotes/inuit)
incoming new (next fetch will store in remotes/inuit)
master new (next fetch will store in remotes/inuit)
v5.0.1 new (next fetch will store in remotes/inuit)
v5.1 new (next fetch will store in remotes/inuit)
v5.1.0 new (next fetch will store in remotes/inuit)
```



Bây giờ bạn có thể chọn một cái remote branche cần fetch dữ liệu về. Ví dụ bây giờ mình sẽ tạo một branch mới cho working tree của mình tên là `fix_ui`, sau đó nạp dữ liệu trong branch `gh-pages` của `inuit` thì mình sẽ lần lượt làm như sau.

```
$ git branch fix_ui
$ git checkout fix_ui
Switched to branch 'fix_ui'
$ git pull inuit gh-pages
warning: no common commits
remote: Counting objects: 309, done.
remote: Compressing objects: 100% (33/33), done.
remote: Total 309 (delta 4), reused 2 (delta 2), pack-reused 274
Receiving objects: 100% (309/309), 135.63 KiB | 0 bytes/s, done.
Resolving deltas: 100% (106/106), done.
From https://github.com/csswizardry/inuit.css
* branch gh-pages -> FETCH_HEAD
* [new branch] gh-pages -> inuit/gh-pages
Auto-merging README.md
CONFLICT (add/add): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
```

Có thể nó sẽ xảy ra lỗi conflict khi gộp file README.md cũng không quan trọng lắm nên bạn có thể bỏ qua bằng cách gõ lệnh `git add README.md` để track file này, vì bây giờ nếu bạn gõ lệnh `ls` ra thì đã có các file từ inuit rồi.

Bây giờ bạn có thể commit nó và thử push nó lên repository của bạn.

```
$ git commit -m "Commit from fix_ui"
[fix_ui cleb301] Commit from fix_ui
$ git push thach fix_ui
Username for 'https://github.com': thachphamblog
Password for 'https://thachphamblog@github.com':
Counting objects: 317, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (201/201), done.
Writing objects: 100% (317/317), 135.77 KiB | 0 bytes/s, done.
Total 317 (delta 113), reused 304 (delta 106)
To https://github.com/thachphamblog/hoc-git.git
* [new branch] fix_ui -> fix_ui
```



Nhắc lại rằng, **thach** nghĩa là tên remote của repository mà mình cần push lên và **fix\_ui** là tên branch mình cần push. Và đây là kết quả sau khi push lên repository của mình.

## Vì dụ kho chứa trong serie Git cơ bản — Edit

 **56** commits

 **3** branches

 **3** releases


 **1** contributor

Your recently pushed branches:

 **fix\_ui** (1 minute ago)

 **Compare & pull request**



 branch: **fix\_ui** ▼


**hoc-git** / +



This b

**Switch branches/tags**



 Pull Request

 Compare

Find or create a branch...

Branches Tags

- ✓ fix\_ui
- master
- version1

		latest commit c1eb30189d
uitcss		a year ago
uitcss		a year ago
		2 years ago
img/content	Add image directory	2 years ago
.gitignore	Add button style and GitHub link	2 years ago
README.md	Commit from fix_ui	6 minutes ago
_config.yml	Fix broken root	2 years ago
atom.xml	Add footer and try RSS	2 years ago
develop.html	Commit develop.html from develop branch	2 hours ago
faq.html	Added faq.html	9 hours ago
icon.png	Add favicon	2 years ago
index.html	Rename from inuit.css to inuitcss	a year ago
inuit.min.css	Add minified v5.0 inuit.css	2 years ago

Và dĩ nhiên, bạn cũng có thể qua một branch khác và sử dụng lệnh `git merge` để gộp dữ liệu của branch này về.

Ngoài ra, kiến thức về branch còn có một kỹ thuật nữa cũng khá thú vị mà khi làm việc nhóm có thể sẽ cần đến, đó là **rebase branch** với mục đích hoán đổi vị trí của những lần commit. Tuy nhiên ở đây là serie cơ bản nên mình sẽ không nói qua mà bạn chỉ cần sử dụng thành thạo các kỹ thuật mà mình đã liệt kê trong đây là được.

## Lời kết

Vậy là tới đây mình đã kết thúc hướng dẫn xong các kỹ thuật quan trọng nhất trong Git mà bạn dù bất cứ sử dụng Git trong mục đích nào cũng phải sử dụng. Mình cũng đã hướng dẫn cặn kẽ nhất theo cách hiểu của mình theo hướng đơn giản nhất nên mình hy vọng bạn có thể hiểu nó. Tiếp tục ở bài sau, mình sẽ không nói qua về các kỹ thuật trong Git nữa mà mình sẽ giới thiệu cho các bạn một số dịch vụ máy chủ repository dành cho Git tốt nhất và đặc biệt là có gói miễn phí để bạn có thêm nhiều lựa chọn.

### Xem tiếp bài trong serie

Phần trước: [\[Git\] Sơ lược Remote Repository và Origin](#)

Phần kế tiếp: [\[Git\] Các dịch vụ repository miễn phí tốt nhất](#)

### Có liên quan

[\[Git\] Git và Github là gì? Tại sao nên dùng?](#)

21/04/2015

Trong "Công cụ"

[\[Git\] Hiểu thêm về Commit và Staging Area](#)

23/04/2015

Trong "Công cụ"

[Làm việc với Git trên WordPress thông qua WP Pusher](#)

12/10/2015

Trong "Wordpress Plugin"

[git](#)[linux](#)

## Thach Pham

Đam mê với web và lập trình, thích viết và chia sẻ, nghiện cà phê và xăm mình, hứng thú với nhạc dân ca và nhạc không lời.

[← Xem thêm bài viết](#)[📡 Subscribe](#)

**BÍ QUYẾT SEO HIỆU QUẢ**  
**Chiến lược SEO lên Top Google trong 1 tháng**  
Dùng Mã **ThachPham** Giảm 500k - [www.dgm.vn](http://www.dgm.vn)

## Để lại bình luận

Hãy là người "bóc tem"

Thông báo cho



Email





[b](#) [i](#) [link](#) [b-quote](#) [u](#) [img](#) [ul](#) [ol](#) [li](#) [code](#)

Bắt đầu thảo luận

**HOSTING 50.000đ**  
LiteSpeed | Free SSL



**AZ**  
**AZDIGI**  
Challenge Accepted

**Đăng ký ngay »**

AZDIGI.com

Bài mới nhất



## Hãy tắt XML-RPC khi không cần sử dụng

🕒 18/09/2016

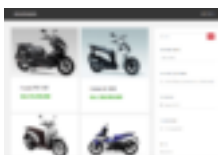
---



## 19 themes WordPress miễn phí đẹp nhất tháng 8 – 9/2016

🕒 18/09/2016

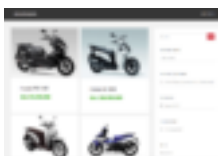
---



## Làm trang shop đơn giản với WordPress – 07

🕒 04/09/2016

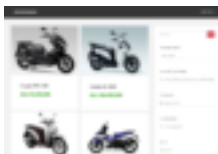
---



## Làm trang shop đơn giản với WordPress – 06

🕒 04/09/2016

---



## Làm trang shop đơn giản với WordPress – 05

🕒 31/08/2016

**DGM**  
Data Marketing Training & Consulting



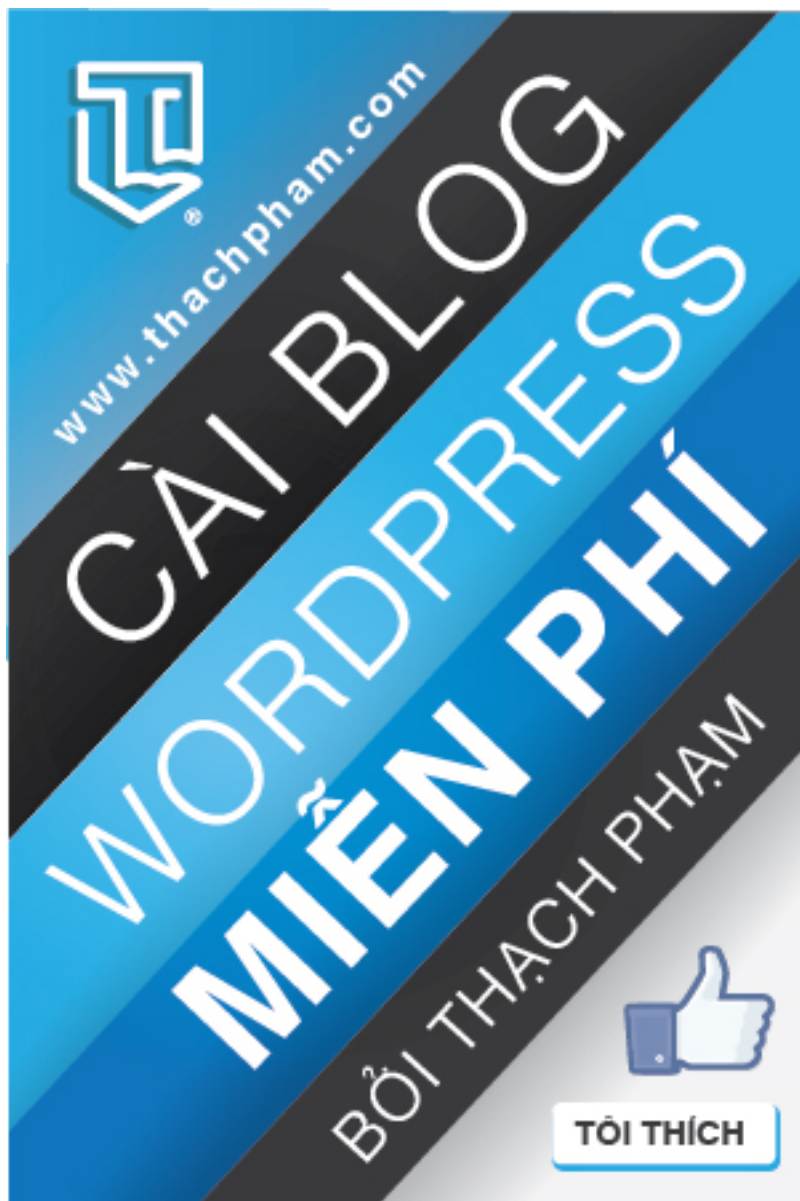
**facebook**

facebook  
**MARKETING**

DÙNG MÃ **THACHPHAM** GIẢM 500K

Bán được hàng **NGAY** trong khóa học





Dành cho người mới



[Học WordPress cơ bản](#)



[Học làm website bán hàng](#)



[Học sử dụng VPS/máy chủ Linux](#)



[Học HTML cơ bản](#)



[Học CSS cơ bản](#)

## Mới cập nhật



### Hãy tắt XML-RPC khi không cần sử dụng



Cập nhật lúc 20/09/2016



### 19 themes WordPress miễn phí đẹp nhất tháng 8 – 9/2016



Cập nhật lúc 18/09/2016



### LEMP cho VPS [Phần 1] – Cài đặt NGINX và PHP-FPM



Cập nhật lúc 17/09/2016

[Giới thiệu](#)

[Tham khảo](#)



ThachPham.Com là blog cá nhân tập trung vào việc phổ biến kiến thức WordPress nói riêng và các kiến thức làm website nói chung đến tất cả mọi người.

Phương châm hoạt động của thachpham.com đó là chia sẻ không giới hạn về những kiến thức bổ ích nhất liên quan đến quản trị website.

## Dịch vụ

---



**Cài đặt WordPress miễn phí**



**Cài đặt máy chủ**



**Thiết kế website WordPress**



**VPS SSD tốc độ cao**



**Tư vấn miễn phí**

- ★ **Các dịch vụ Shared Host tốt nhất**
- ★ **Các dịch vụ VPS tốt nhất**
- ★ **Nên mua theme WordPress ở đâu?**

## Liên kết

---

- ▶ [Trần Ngọc Chính](#)
- ▶ [IZWebz](#)
- ▶ [Vietdesigner](#)
- ▶ [Danny Nguyễn](#)
- ▶ [Ngân Sơn](#)
- ▶ [Văn Mỹ](#)
- ▶ [Lập trình Sen Việt](#)
- ▶ [Trung Đức](#)
- ▶ [Vinacode](#)
- ▶ [Giúp Bạn Làm Đẹp](#)

- ▶ [Lại Văn Đức](#)
- ▶ [Ninh Đôn](#)
- ▶ [Đinh Trang](#)

---

Xin vui lòng trích nguồn **Thach Pham Blog** khi phát hành lại nội dung trên website này.  
Giao diện thiết kế bởi [orange-themes.com](#)