**.GITTIGNORE**

**1.Gittignore là gì?**

Gitignore là file có tên là .gitignore do Git quy định. Nhiệm vụ của nó là liệt kê những file mà mình không mong muốn cho vào git hoặc hiểu nôm na là Git sẽ bỏ qua những file đó đi.

**2.Gittignore hoạt động như thế nào?**

Git sẽ quản lý các file chúng ta muốn ignore bằng đuôi .gitignore được đặt tại root của project. Khi có một sự thay đổi trong Git, file thay đổi đó được đưa vào Git Cache để phân tích cú pháp. Vì thế người dùng phải đặt lệnh đẩy nó ra khỏi Git Cache thì .gitignore mới có thể hoạt động bỏ qua.

Những file cần bỏ qua ở đây có thể là

-File IDE tự động sinh ra,

-Các file để chạy ứng dụng của máy cá nhân

-Các file cá nhân cần bảo mật của bản…

Việc sử dụng Gitignore sẽ giúp hạn chế:

- Hao phí tài nguyên

-Bộ nhớ cho server

-Tránh trường hợp xung đột giữa các phần mềm từ máy cá nhân của thành viên trong team hoặc lộ dữ liệu cần bảo mật trong repository.

**3.Khi nào nên sử dụng gittignore file?**

Bất cứ project nào cũng nên dùng nó, nên tạo ngay file .gitignore trong thư mục gốc ngay khi khởi tạo project của bạn và liệt kê luôn những file mà bạn muốn git bỏ qua.

**Tại sao phải liệt kê trước làm gì thế?**

Chú ý – Git cache

Giả dụ thế này! Bạn vừa join vào project và thấy project liên tục bị conflict vì mấy file rác. Nhưng may quá bạn đọc được bài viết này và bạn rất thông minh nên đã tạo luôn file .gitignore cho project và thêm luôn file rác đó vào .gitignore rồi bạn xóa file rác đi và commit lên.

Rồi sao! 1 ông khác lại pull code mới về lại tạo ra file rác đó và nó vẫn dính vào git bình thường. Đờ heo? “Em cho nó vào .gitignore rồi cơ mà?.

Vì sao à? Vì file đó đã được thằng git cache thu nạp thành của nó rồi nên thằng git nó vẫn có quyền quản lý file đó. Vậy cách giải quyết đơn giản nó phải giải thoát file đó ra khỏi git cache là xong, bằng 1 dòng lệnh:

git rm -r --cached /path/to/file\_or\_folder

Từ bây giờ file đó không còn là của **git cache** nữa nên nó không thuộc quyền quản lý của git nữa và bây giờ **.gitignore** mới có tác dụng. Theo lý thuyết là vậy nhưng nếu bạn cần reset lại hết project để **.gitignore** hoạt động đúng thì mình thường xóa bỏ hết file của **git cache** luôn?

git rm -r --cached .

Sau đó mình **sẽ add tất cả các file** lại vào project như lúc mới tạo project.

git add

Và bây giờ bạn lại **commit** và **push** như bình thường.

**4.Nên dùng Ignore File Git cho những đối tượng nào?**

Thực ra thì việc quyết định dùng Gitignore cho tệp tin nào còn tùy thuộc vào nhu cầu bản thân người dùng. Tuy nhiên, đây sẽ là những đối tượng hay được sử dụng nhất:

-Các file hệ thống

-Các file cấu hình

-Các phải phát sinh trong quá trình biên dịch

-File lưu trữ và tìm kiếm các module

-File văn bản và dữ liệu cá nhân

-Cơ sở dữ liệu và các file lưu log

**5.Đặc điểm của Gitignore File là gì?**

**Tầm ảnh hưởng**

Khi ta add Git Ignore vào trong repo, nó sẽ mang đến tầm ảnh hưởng như thế nào? Khi nhận được những thông tin gợi tả về các dữ liệu cần ignore, Git sẽ quyết định thứ tự ưu tiên bỏ qua từ cao nhất đến thấp nhất. Cao nhất và thấp nhất ở đây để chỉ các nguồn. Những file có liên hệ với nhiều nguồn mà Gitignore sẽ được bỏ qua đầu tiên.

Vậy nếu hai file có cùng số nguồn thì sao? Trong trường hợp như vậy, thường 2 file này là tệp lồng nhau. Tức là có 1 folder lớn và những file nhỏ ở trong. Nếu đã như vậy, Gitignore sẽ ưu tiên bỏ qua folder to nhất trước.

Tuy nhiên đặc điểm riêng biệt của Gitignore là chỉ ảnh hưởng được đến những file cùng bậc hoặc nhỏ hơn. Vì thế Gitignore File thường được đặt ở thư mục root để có đủ quyền hạn bao quát cả project.

**Cú pháp**

Để Gitignore có thể nhận dạng chuẩn xác được những File bạn cần bỏ qua đòi hỏi lúc bạn nhập cú pháp phải thật chuẩn.

Trong Git Ignore Folder không tồn tại cú pháp trống bởi file nào cũng chứa phần trống. Vì thế tác dụng của nó chỉ là phần cách cho dễ đọc thôi.

-Nếu muốn add thêm chú thích vào cú pháp, bạn có thể dùng # để mở đầu bình luận đó.

-Dấu gạch chéo “/” phải được đặt ở cuối cú pháp. Như vậy thì nó mới được nhận định là một Gitignore. Nếu không nó có thể được đọc là một file thường hoặc một symbolic link.

-Dấu chấm than “!” có nghĩa là phủ định. Do vậy nên nếu bạn thêm ! vào trước một file, nó sẽ không bị bỏ qua nữa mà được thêm vào lại. Tuy nhiên nếu một folder lớn đã bị bỏ qua thì cho dù có thêm ! vào các thư mục con ở trong nó thì cũng không thể được thêm vào lại.

-Hai dấu sao đúp “\*\*” được dùng cho trường hợp không muốn hoặc không thể định rõ tên. Ví dụ nếu bạn sử dụng cú pháp \*\*/abc, nó sẽ có hiệu lực với tất cả các file có chứa abc trong tên. Còn nếu bạn sử dụng dạng abc/\*\* thì nó sẽ có hiệu lực với tất cả các tệp con trong thư mục abc.