1. Giới thiệu

Dưới đây là những phương pháp phát triển phần mềm mà C9js đã sử dụng:

* 1. Tích hợp liên tục (Continuous Integration)
     1. Khái niệm

Theo Martin Fowler, tích hợp liên tục là phương pháp phát triển phần mềm đòi hỏi các thành viên trong nhóm tích hợp công việc thường xuyên. Mỗi ngày, các thành viên đều phải theo dõi và phát triển công việc của họ ít nhất một lần. Việc này sẽ được một nhóm khác kiểm tra tự động, nhóm này sẽ tiền hành kiểm thử truy hồi để phát hiện lỗi nhanh nhất có thể. Cả nhóm thấy rằng phương pháp tiếp cận này giúp giảm bớt vấn đề về tích hợp hơn và cho phép phát triển phần mềm gắn kết nhanh hơn.

* + 1. Điều kiện áp dụng

Những dự án được hưởng lợi từ tích hợp liên tục:

* Nhóm phát triển có ít hơn 50 người, độ phức tạp của dự án không cao
* Các sản phẩm có sử dụng phần mềm nhúng
  + 1. Lợi ích

Một số lợi ích mà tích hợp liên tục mang lại:

* Giảm thiểu rủi ro vì có thể phát hiện sớm lỗi, tiết kiệm thời gian và tiền bạc khi sửa lỗi
* Hạn chế lỗi khi sắp đến ngày bàn giao sản phẩm
* Nhà phát triển giảm thời gian sửa lỗi khi phát hiện lỗi lúc chạy kiểm thử đơn vị do họ thường xuyên tích hợp
* Luôn có sản phẩm có thể chạy hoặc demo cho khách hàng
  1. Chuyển giao liên tục (Continuous Delivery)
     1. Khái niệm

Theo Martin Fowler, Chuyển giao liên tục là nguyên tắc phát triển phần mềm mà ở đó phần mềm có thể được phát hành ở bất kì thời gian nào.

* + 1. Điều kiện áp dụng

Nên áp dụng chuyển giao liên tục khi:

* Phần mềm có thể triển khai xuyên suốt vòng đời của nó
* Ưu tiên việc triển khai hơn là tạo ra tính năng mới
* Mọi người có thể lấy được phản hồi nhanh chóng và tự động khi có người thay đổi hệ thống
* Có thể triển khai bất kì phiên bản ở bất kì môi trường nào theo nhu cầu.
  + 1. Lợi ích

Các lợi ích chủ yếu của chuyển giao liên tục:

* Giảm rủi ro khi triển khai: khi người phát triển triển khai những thay đổi nhỏ, khi đó khả năng sai sót sẽ ít hơn và dễ dàng sửa lỗi khi trục trặc phát sinh
* Tiến trình trở nên tin cậy: những nhà phát triển theo dõi tiến trình bằng cách theo dõi công việc khi hoàn thành. Nếu “hoàn thành” tức là “nhà phát triển sẽ thông báo là nó được hoàn thành”, điều này có ít độ tin cậy hơn việc triển khai nó thành sản phẩm
* Phản hồi người dùng: rủi ro lớn nhất đối với bất kì phần mềm nào đó là nhà phát triển cho ra một thứ nào đó không hữu dụng. Nếu nhà lập trình làm việc với người dùng thật sớm hơn và thường xuyên hơn, thì họ sẽ có được phản hồi nhanh chóng hơn để tìm ra giá trị thực sự của phần mềm

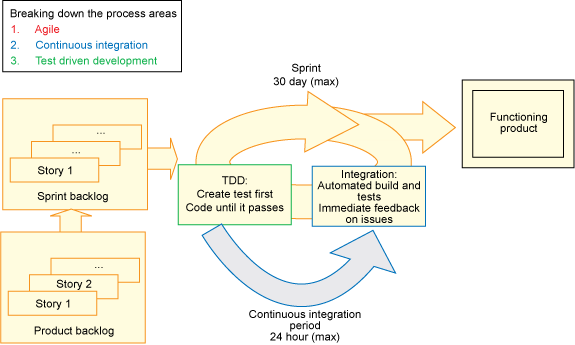
1. Travis CI
   1. Giới thiệu

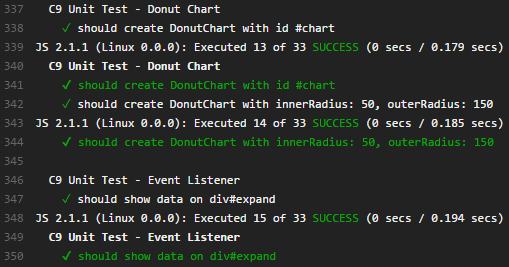
Travis CI là hệ thống hiện thực tích hợp liên tục (CI) nhanh chóng và gọn lẹ, tích hợp với Github, GitLab, Bitbucket.

* 1. Các tính năng
* Theo dõi các test khi các test này chạy
* Giữ các cấu hình (config) cùng với mã nguồn của nhà phát triển
* Hỗ trợ thông báo qua các kênh: Slack, HipChat, Email,...
* Một máy ảo hoàn toàn mới mỗi khi biên dịch (build)
* Chạy các test song song
* Hỗ trợ thêm các hệ điều hành Linux, Mac (và iOS)
* Có command line và API mạnh mẽ
* Hoàn toàn miễn phí cho các phần mềm mã nguồn mở
  1. Ứng dụng

Mọi việc trở nên tự động khi sử dụng Travis CI. Mỗi khi commit mã nguồn, hệ thống sẽ tự động lấy mã từ nguồn tích hợp, biên dịch, sau đó chạy các bộ kiểm thử cũng như thông báo tình trạng hiện tại của dự án qua nhiều kênh: email, Slack, ... từ đó dễ dàng tìm ra “hung thủ” làm hư hại mã, cũng như các test không vượt qua được,... Sau khi hệ thống chạy thành công, có thể yêu cầu thực hiện một vài tác vụ như báo cáo độ bao phủ mã nguồn, phát hành phần mềm,...

C9js áp dụng Travis CI để hiện thực tích hợp liên tục, phát triển hướng kiểm thử, cùng với chuyển giao liên tục trong quá trình phát triển thư viện. Qui trình thực hiện sẽ được nêu rõ trong phần *Hiện thực thư viện*.

  
Sự kết hợp giữa tích hợp liên tục (CI) và phát triển theo hướng kiểm thử (TDD)



Chạy kiểm thử đơn vị bởi Travis CI