# Giới thiệu SignalR

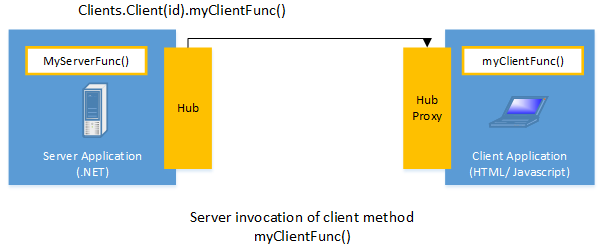
## SignalR là gì?

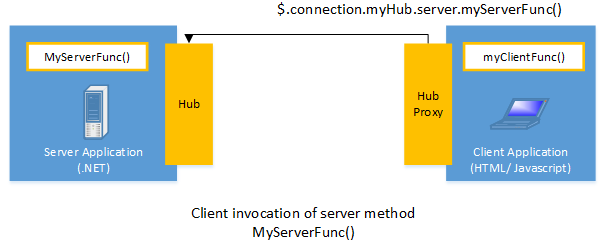
ASP.NET SignalR là một thư viện dành cho các developer ASP.NET giúp đơn giản hóa quá trình thêm các chức năng web thời gian thực vào các ứng dụng. Chức năng web thời gian thực là server có khả năng đẩy nội dụng đến các client đã được kết nối ngay lập tức khi chúng kết nối sẵn với server. Thay vì máy chủ chờ một client gởi dữ liệu request mới.

SignalR có thể được dùng thêm bất kỳ loại chức năng web “thời gian thực” cho ứng dụng ASP.NET của bạn. Trong khi chat thường được sử dụng làm ví dụ, thật ra bạn có thể làm được nhiều hơn thế. Bất kỳ lúc nào người dùng refreshes một trang web để xem dữ liệu mới, hoặc các trang thực hiện long polling (Client sẽ gửi request đến server, server tiến hành kiếm tra dữ liệu và đến khi nào có dữ liệu mới thì mới gửi response về cho client. Sau đó client lại tiếp tục tự động gửi 1 request mới và đợi dữ liệu mới trả về.) để nhận dữ liệu mới, nó là một đại diện dùng SignalR. Các ví dụ khác bao gồm các dashboard và các ứng dụng giám sát, các ứng dụng có sự hợp tác (chẳng hạn như chỉnh sửa đồng thời các tài liệu), cập nhật tiến độ công việc, và các form thời gian thực.

Signal cũng cho phép tạo ra các loại ứng dụng web hoàn toàn mới yêu cầu tần xuất cập nhật cao từ server, ví dụ, game thời gian thực. Ví dụ tuyệt với cho trường hợp này là ShootR game.

Signal cung cấp một API đơn giản cho tạo thủ tục server-to-client từ xa gọi là RPC. Nó gọi các hàm JavaScript ở các browers client (và các nên tảng client khác) đến phía server .NET code. SignalR cũng thêm API cho quản lý kết nối (cho instance, sự kiện connect và disconnect) và nhóm các kết nối.





SignalR thực hiện quản lý kết nối một cách tự động, và cho phép bạn phát đi các tin nhắn cho tất cả các client kết nối cùng một lúc, giống một chat room. Bạn cũng có thể gởi các thông điệp đến các clients cụ thể. Kết nối giữa client và server là liên tục, không giống như một kết nối HTTP cơ bản, được thiết lập lại cho mỗi lần giao tiếp.

SignalR hỗ trợ chức năng “server push”, trong đó code server có thể gọi đến code của client trên browser dùng Remote Procedure Calls (RPC). Thay vì mô hình request-response phổ biến trên web hiện nay.

Các ứng dụng SignalR có thể mở rộng ra hàng ngàn client dùng Service Bus, SQL Server hoặc Redis

SignalR là open-source, có thể truy cập thông qua GitHub

## SignalR và WebSocket

SignalR dùng WebSocket (chỉ hỗ trợ trên HTLM5) để vận chuyển các thông tin. Khi bạn có thể viết ứng dụng của bạn dùng trực tiếp WebSocket, việc dùng Signal có ý nghĩa rằng nhiều tính năng mà bạn sẽ phải hiện thực SignalR đã làm hết cho bạn. Quan trọng nhất, điều này có nghĩa bạn có thể code chương trình của bạn tận dụng WebSocket mà không phải lo lắng về tạo các mã riêng cho các client sử dụng công nghệ cũ không phải HTML5. SignalR cũng giúp bạn khỏi phải lo lắng vì việc cập nhật của WebSocket. Vì SignalR sẽ tiếp tục được cập nhật để hỗ trợ sự thay đổi trong vận chuyển dữ liệu bên dưới. Cung cấp cho ứng dụng của bạn một giao diện nhất quán trên các phiên bản của WebSocket.

Mặc dù bạn chắc chắn có thể tạo ra một giải pháp sử dụng mỗi WebSocket, SignalR cung cấp tất cả các tính năng bạn sẽ cần để viết code, chẳng hạn như dự phòng cho các phương thức vận chuyển khác và sửa đổi ứng dụng của bạn khi triển khai các bản cập nhật WebSocket.

## Các vận chuyển và dự phòng

SignalR là một sự trừu tượng hóa với một số các vận chuyển được yêu cầu làm việc thời gian thực giữa client và server. Một kết nối SignalR bắt đầu như HTTP, và sau đó được đẩy lên một kết nối WebSocket nếu có sẵn. WebSocket là phương tiện lý tượng để vận chuyển cho SignalR, vì nó làm tối ưu việc sử dụng bộ nhớ máy hiệu quả nhất, độ trễ thấp nhất, và có nhiều tính năng ngầm nhất (như truyền thông song song giữa client và server), nhưng nó cũng có những yêu cầu nghiêm ngặt nhất: WebSocket yêu cầu server dùng Windows Server 2012 hoặc Windows 8, và .NET Framework 4.5. Nếu những yêu không đó không được đáp ứng, SignalR sẽ cố gắng các phương thức vận chuyển khác để kết nối.