Giới thiệu ASP.NET vNext and MVC6

Một trong những điểm mới của ASP.NET vNext, ASP.NET MVC 6 đại diện cho sự thay đổi cơ bản về cách xây dựng và triển khai web frameworks. Điều này có mục đích chính là để tạo ra một host agnostic framework không còn phụ thuộc vào cơ sở System.Web.

Microsoft thấy rằng System.Web cần phải được loại bỏ bởi vì nó thực sự khá tốn kém. Một HttpContext điển hình có thể tiêu thụ 30K của bộ nhớ cho mỗi yêu cầu. Khi nó làm việc với các yêu cầu loại JSON nhỏ thì lại không tương xứng. Với thiết kế mới của MVC 6, yêu cầu đầu tiên giảm xuống 2K bộ nhớ.

Trong MVC 6 là Web API và Web Pages, Microsoft cho phép loại bỏ nhiều sự trùng lặp giữa ba framework này. Một trong những kết quả của sự thay đổi này là MVC sẽ tự hosting giống như API 2 và SignalR 2.

Để làm cho việc triển khai dễ dàng và đáng tin cậy hơn, “vNext sẽ hỗ trợ triển khai thực side-by-side”. Thay vì được cài đặt trong GAC, mỗi thư viện MVC cần thiết cho một trang web sẽ được tham chiếu giống như developer tạo ra DLL bình thường.”Điều này có nghĩa là bạn có thể cập nhật ứng dụng của bạn mà không ảnh hưởng đến các ứng dụng khác trên cùng một máy chủ”.

## Pay as you go

MVC 6 được xây dựng trên một triết lý “pay as you go”. Mỗi tính năng mà bạn muốn sử dụng phải được bật trên ứng dụng khởi động. Thậm chí, phục vụ file tĩnh yêu cầu gọi IBuilder.UseStaticFiles.

Cách làm việc này là mỗi trang web cần phải có một lớp tên Startup và lớp này phải có phương thức “void Configure (IBuilder app)”. Bên trong phương thức này bạn có thể gọi hàm như trong “app.UseServices” để chạy các tính năng như MVC.

## Triển khai Azure và PowerShell Based

Microsoft vẫn dùng Azure như là tiêu chuẩn để triển khai xây dựng các websites. Nhưng họ vẫn thấy các nhà phát triển tạo các websites từ Visual Studio. Vì vậy thay vào đó họ sẽ tạo ra đoạn PowerShell scripts theo mặc định. Tiếp đó, chúng có thể được chỉnh sửa bên trong Visual Studio, mà bây giờ đã có công cụ cơ bản hỗ trợ cho PowerShell.

## Các Build Process không xây dựng

Trong ASP.NET vNext quá trình xây dựng không tạo ra bất cứ điều gì. Không có chương trình được tạo ra, nó chỉ chạy kiểm tra để đảm bảo rằng bạn không có bất kì lỗi nào hoặc cảnh báo rằng cần được giải quyết. Thay vào đó, các đoạn mã cần được biên dịch, giống như các trang ASP.NET chúng ta vẫn thường thấy. Điều này cho phép lặp lại nhanh hơn, đặc biệt là trên các trang web lớn.

Nếu bạn muốn các chương trình thực được triển khai trên một máy chủ bạn cần chạy các package và lệnh. Cuối cùng điều này sẽ cung cấp một số tùy chọn từ mã nguồn, nó sẽ tiếp tục để biên dịch. Điều cuối nên có hiệu suất tốt hơn nhưng nó có thể đòi hỏi một quá trình lâu hơn.

## Nhiều APIs sẽ bị chuyển hoặc loại bỏ

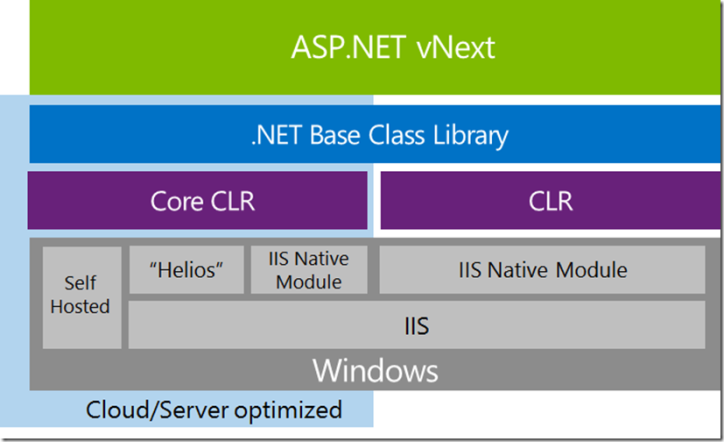
Như đã đề cập ở trước, họ đang giảm kích thước của HttpContext từ khoảng 30K xuống 2K cho một yêu cầu. Đây không phải là miễn phí, chi phí để làm điều này là giảm đáng kể cài đặt phương thức trên các đối tượng. Và khi chúng được thực hiện, nó có lẽ không chỉ có API bị cắt giảm kích thước.

Để thực hiện sự thay đổi này gây ít tổn thất, họ có ý định phát triển một FxCop như công cụ mà nó có sẽ phát hiện khi cuộc goi APIs đang được thực hiện. Trong khi nó không thể tự viết lại mã của bạn, ít nhất thì nó cũng có thể cho bạn biết những gì cần phải thay đổi trước khi chuyển sang ASP.NET vNext và MVC 6.

## Framework đầy đủ và Cloud Optimized Framework

Nếu bạn thực hiện các bước tiếp theo với những gì họ gọi là “Cloud Optimized Framework” thì sau đó bạn sẽ mất quyền truy cập tới tất cả những API. Trong Channel 9 Q&A họ nói System.Drawing là một ví dụ bạn không thể sử dụng được.

Ưu điểm của việc sử dụng Cloud Optimized Framework là bạn có thể gộp một bản sao của Core (or Mono) CLR với trang website của bạn. Bạn không cần phải nâng cấp .NET trên toàn bộ máy của một website. Thậm chí bạn có thể có các phiên bản khác của CLR cho các trang web đang cùng chạy.



*Hình 1: ASP.NET vNext*

## Thư viện và package

Theo mô hình .NET vNext, project không có bất cứ thư viện riêng nào. Thay vào đó họ sử dụng NuGet Packages. Như bạn đã biết, package có thể chứa nhiều phiên bản của thư viện đã phân theo platform. ASP vNext có thể tận dụng điều này để trong thời gian chạy có thể load Full .NET, Mono, or Core CLR phiên bản nào của thư viện được.

## Mono là một Supported Platform

Trước đây, câu chuyện hỗ trợ của Mono “Chúng tôi hy vọng nó chạy nhưng nếu nó không chạy thì bạn cần nói cho Xarmarin”. Ngày nay, Microsoft đang coi Mono như cross-platform CLR chính dành cho ASP.NET vNext. Họ đang tích cực làm việc với các đội Mono để đảm bảo nó có mọi thứ họ cần và sẽ có Mono trong thử nghiệm Continuous Integration của họ.

Điều đó nói lên rằng, Microsoft không cung cấp hỗ trợ chính cho Mono. Họ chỉ hứa duy trì khả năng tương thích và nếu một bài test CI lỗi thì họ sẽ làm việc với nhóm Mono để sửa nó.

## Phát triển Platform Development

Microsoft không chỉ đang có kế hoạch cho việc triển khai cross-platform, họ cũng được cho phép phát triển cross-platform. Batch file dành cho các nền tảng lớn như OS X và Linux sẽ được cung cấp để bạn có thể đóng gói và triển khai các dự án ASP.NET vNext mà không cần Windows và Visual Studio.