

# Blazor – client web app với C# (và không JavaScript)

Hướng dẫn tự học lập trình ASP.NET Core toàn tập > Blazor – client web app với C# (và không JavaScr...

Blazor là framework mới nhất trong gia đình Asp.net Core giúp phát triển ứng dụng đơn trang (Single-Page Application, SPA) nhưng sử dụng ngôn ngữ C#. Blazor cũng là framework đang nhận được sự quan tâm lớn từ cả Microsoft và cộng đồng. Đây hứa hẹn sẽ là một framework trọng tâm trong tương lai của Asp.net Core.

Bắt đầu từ bài học này chúng ta sẽ chuyển sang xem xét cách làm việc với framework này.

Do đặc thù là một framework rất mới và đang trong quá trình phát triển, nội dung của phần này có thể sẽ không được đầy đủ và chi tiết như hai framework còn lại (Razor pages, MVC/API). Chúng tôi sẽ cố gắng cập nhật các nội dung của phần này khi có thể.

## NỘI DUNG CỦA BÀI [ Ấn ]

1. Blazor là gì?
2. Tạo project Blazor đầu tiên
3. Hello world, Blazor server!
4. Dự án Blazor WebAssembly
5. Kết luận
  - 5.1. Tải mã nguồn Blazor

## Blazor là gì?

Blazor là một framework miễn phí mã mở và là một bộ công cụ mới của Microsoft giúp phát triển client web (chạy trên trình duyệt) sử dụng ngôn ngữ C# (thay cho JavaScript truyền thống). Bản release chính thức của Blazor mới phát hành vào tháng 9/2019 cùng với Asp.net Core 3.0.

Trước hết cần khẳng định rằng đây là viết client web app bằng C# thực sự, không dùng JavaScript, cũng không phải là dùng một transpiler (công cụ dịch mã C# thành mã JavaScript) như trước kia. Blazor giúp bạn tạo ra ứng dụng .NET thực sự chạy trên browser mà không cần thêm plugin hay add-on nào hết.

Trước đây một số công cụ cũng cho phép viết code C# cho trình duyệt, ví dụ <https://bridge.net/>. Tuy nhiên, các công cụ này dịch mã nguồn C# sang JavaScript. Nói cách khác, bản chất của các ứng dụng đó cũng là ứng dụng JavaScript. Việc dịch mã nguồn các ngôn ngữ khác sang JavaScript như vậy gọi là transpilation. Công cụ để dịch mã như thế gọi là transpiler.

Nếu bạn là một lập trình viên .NET, bạn hoàn toàn không cần rời xa ngôn ngữ **lập trình C#** và môi trường Visual Studio quen thuộc. Bạn cũng không cần cài đặt hàng loạt thư viện hỗ trợ.

Với Blazor, bạn hoàn toàn có thể viết ra các ứng dụng web client native, giống hệt như những gì mà Angular hay React thực hiện được. Tất cả chỉ với C# và Visual Studio.

Trước đây nếu cần tạo ra giao diện tương tác trên web bạn chỉ có lựa chọn là JavaScript và các framework của nó. Giờ đây, Blazor cũng có thể tạo ra UI tương tác hoàn toàn tương tự từ HTML, CSS và C#!!! Cả client và server giờ có thể viết bằng C#.

Các đặc điểm quan trọng của Blazor là:

- Blazor được xây dựng trên các tiêu chuẩn của web mở mà bạn đã quen thuộc.
- Blazor cung cấp khả năng giống như của các JavaScript framework thông dụng như Angular hay React.
- Blazor cung cấp khả năng tương tác với JavaScript. C# code có thể gọi JavaScript code để tận dụng các thư viện JavaScript khổng lồ!
- Khác với người tiền nhiệm SilverLight, Blazor không đòi hỏi trình duyệt phải cài đặt thêm plugin. Blazor cũng không phải là một loại transpiler và cũng không sử dụng bất kỳ transpilation nào.

Blazor được phát triển làm một bộ phận của Asp.net Core, cùng hoạt động song song với MVC và Razor page đã có từ trước. Do vậy, những kinh nghiệm và kỹ thuật đã biết với nền tảng này có thể tiếp tục được tận dụng với Blazor. Blazor cũng có khả năng tái sử dụng code và thư viện như các ứng dụng .NET.

Mặc dù mới ra đời, các hãng thứ ba nổi tiếng như Telerik, DevExpress, Syncfusion đã bắt đầu cung cấp thư viện component cho Blazor. Bạn có thể truy cập để dùng thử miễn phí.

Hiện nay Blazor có hai mô hình hoạt động chính: (1) chạy trên server, gọi là **Blazor Server**; (2) chạy trên WebAssembly, gọi là **Blazor Assembly**. Sự khác biệt này chúng ta sẽ xem trong bài viết tiếp theo. Ngoài ra, một số mô hình Blazor khác vẫn đang trong quá trình phát triển.

## Tạo project Blazor đầu tiên

---

Để làm việc với Blazor, bạn cần phiên bản Visual Studio 2019 **tối thiểu 16.3.6** với **ASP.NET and web development** workload.

Để kiểm tra phiên bản của Visual Studio, bạn mở Visual Studio Installer. Nếu chưa update lên phiên bản 16.3.6, hãy thực hiện update lên phiên bản mới nhất.

Nếu chưa cài ASP.NET and web development workload, hãy click nút Modify và cài đặt như dưới đây.

Lý do phải update là bản 16.3.6 lần đầu được tích hợp với .NET Core 3.0 – phiên bản đầu tiên hỗ trợ Blazor Server app.

# Visual Studio Installer

Installed

Available



Visual Studio Community 2019

16.3.6

Powerful IDE, free for students, open-source contributors, and individuals

[Release notes](#)

Modify

Launch

More ▾

Modifying — Visual Studio Community 2019 — 16.3.6

Workloads

Individual components

Language pack

Web & Cloud (4)



ASP.NET and web development

Build web applications using ASP.NET Core, ASP.NET, HTML/JavaScript, and Containers including Docker support.



Giờ bạn có thể bắt đầu tạo project cho ứng dụng Blazor:

## Create a new project

Recent project templates

Class Library (.NET Framework)

C#

Console App (.NET Framework)

C#

Blank Solution

Blazor

Clear

All Languages

All Platforms

All Project Types



Blazor App

Project templates for creating Blazor apps that run on the server in an ASP.NET Core app or in the browser on WebAssembly. These templates can be used to build web apps with rich dynamic user interfaces (UIs).

C#

Linux

macOS

Windows

Cloud

Web

## Configure your new project

Blazor App

C#

Linux

macOS

Windows

Cloud

Web

Project name

HelloBlazor

Location

\TuHocICT\Blog\Code\

Solution name ⓘ

HelloBlazor

☐ Place solution and project in the same directory

# Create a new Blazor app

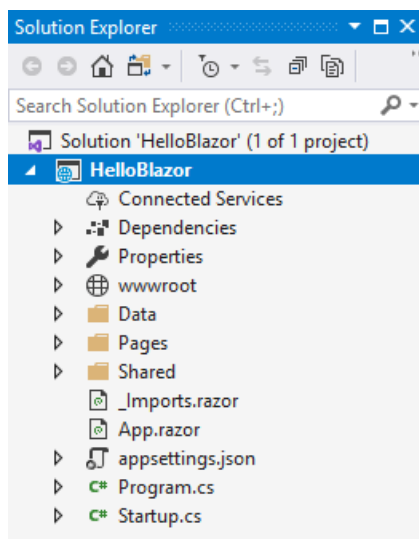
ASP.NET Core 3.0

**Blazor Server App**  
A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This template can be used for web apps with rich dynamic user interfaces (UIs).

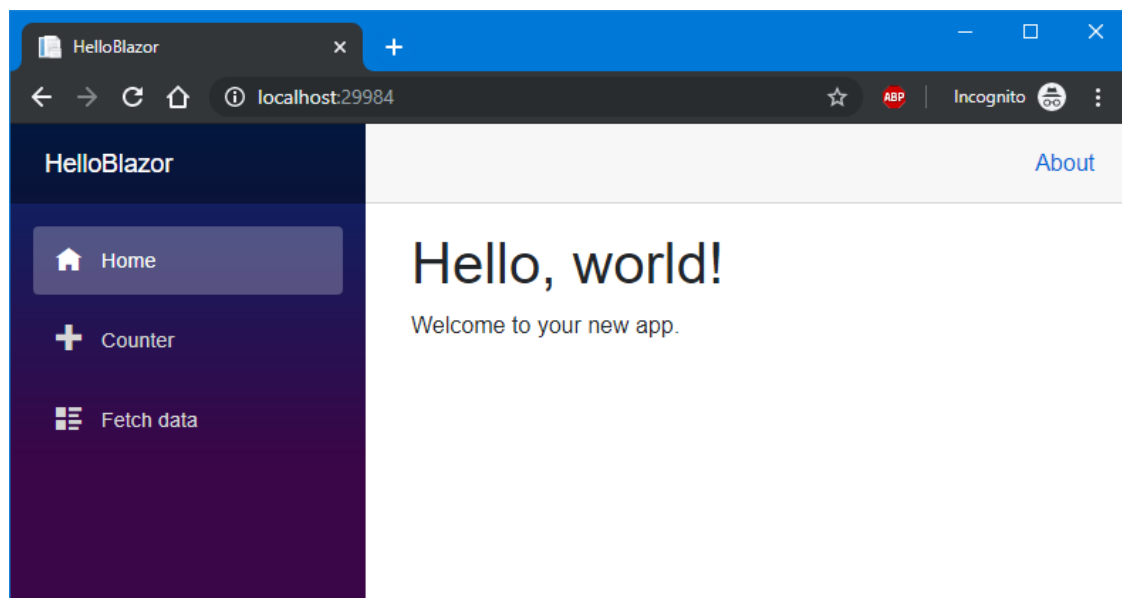
**Authentication**  
No Authentication  
[Change](#)

**Advanced**  
☐ [Configure for HTTPS](#)  
☐ [Enable Docker Support](#)  
(Requires [Docker Desktop](#))

Khi hoàn thành, bạn sẽ thu được một project với cấu trúc như sau:

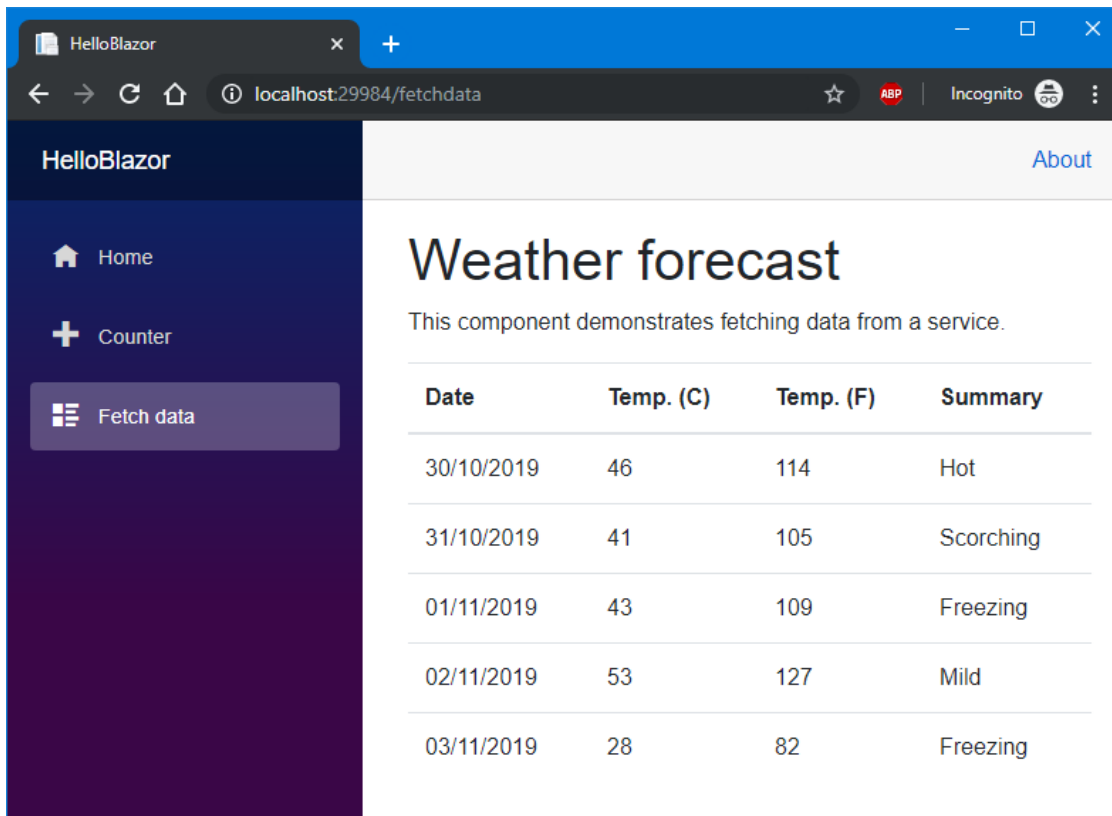
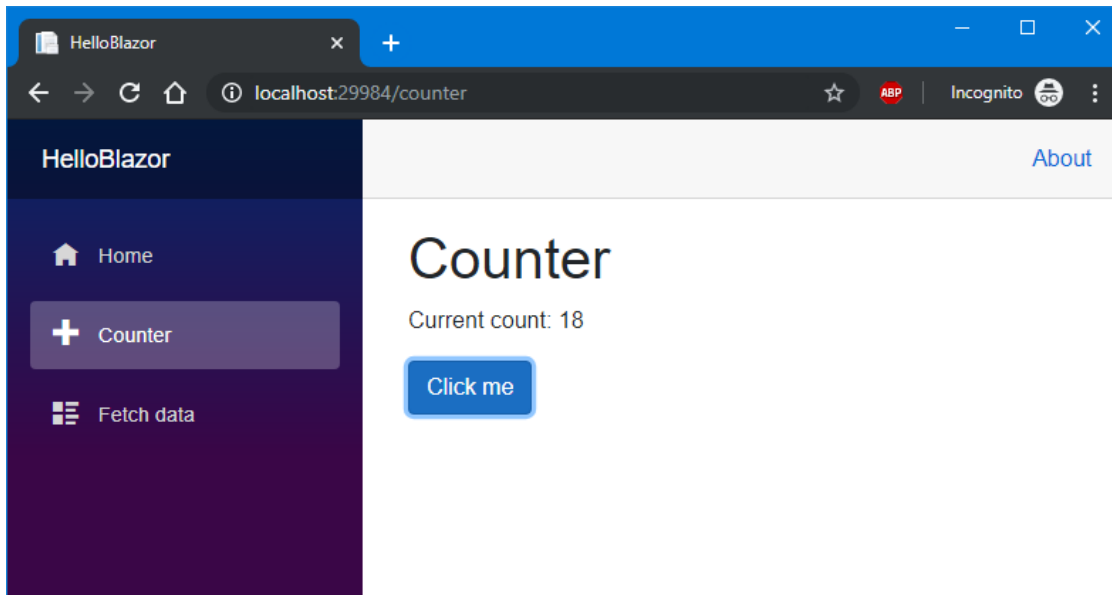


Dịch và chạy chương trình, bạn sẽ thu được ngay một ứng dụng web như sau:



Đây là ứng dụng mẫu mà Blazor tạo sẵn, giúp bạn có một cái khung để dễ dàng phát triển ứng dụng của riêng mình.

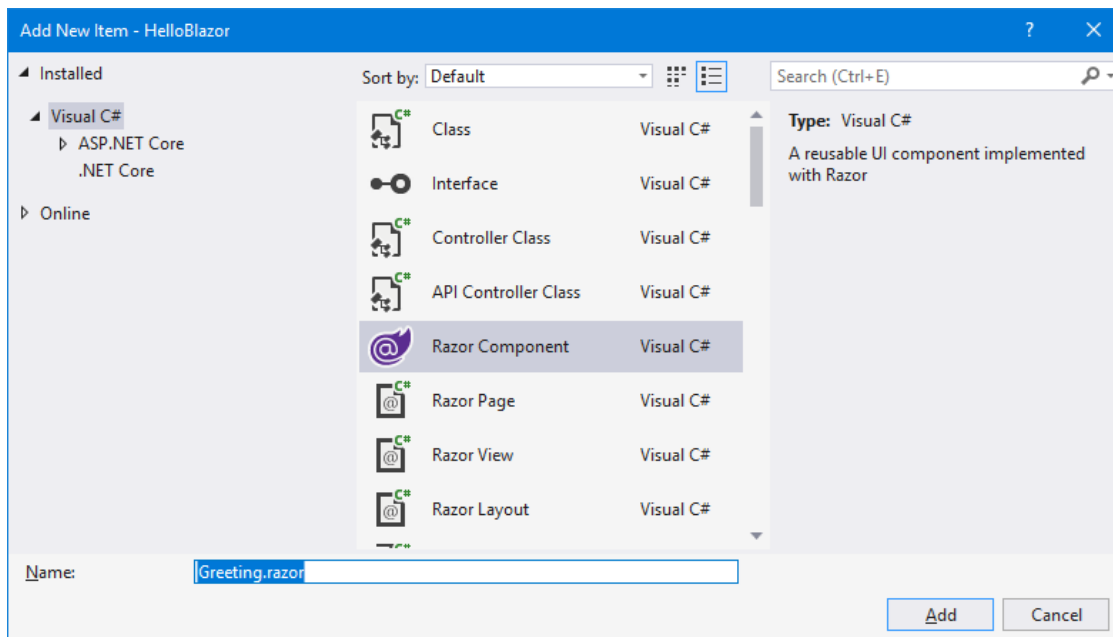
Hãy dành chút thời gian để nghịch với chương trình đơn giản này:



## Hello world, Blazor server!

Chúng ta tiếp tục với việc xây dựng một page riêng để cảm nhận cách viết code trong Blazor.

Click phải vào folder Pages và chọn Add => New Item. Sau đó chọn **Razor Component** đặt cho nó một cái tên (Greeting.razor).



Giờ là lúc viết vài dòng code cho Greeting.razor:

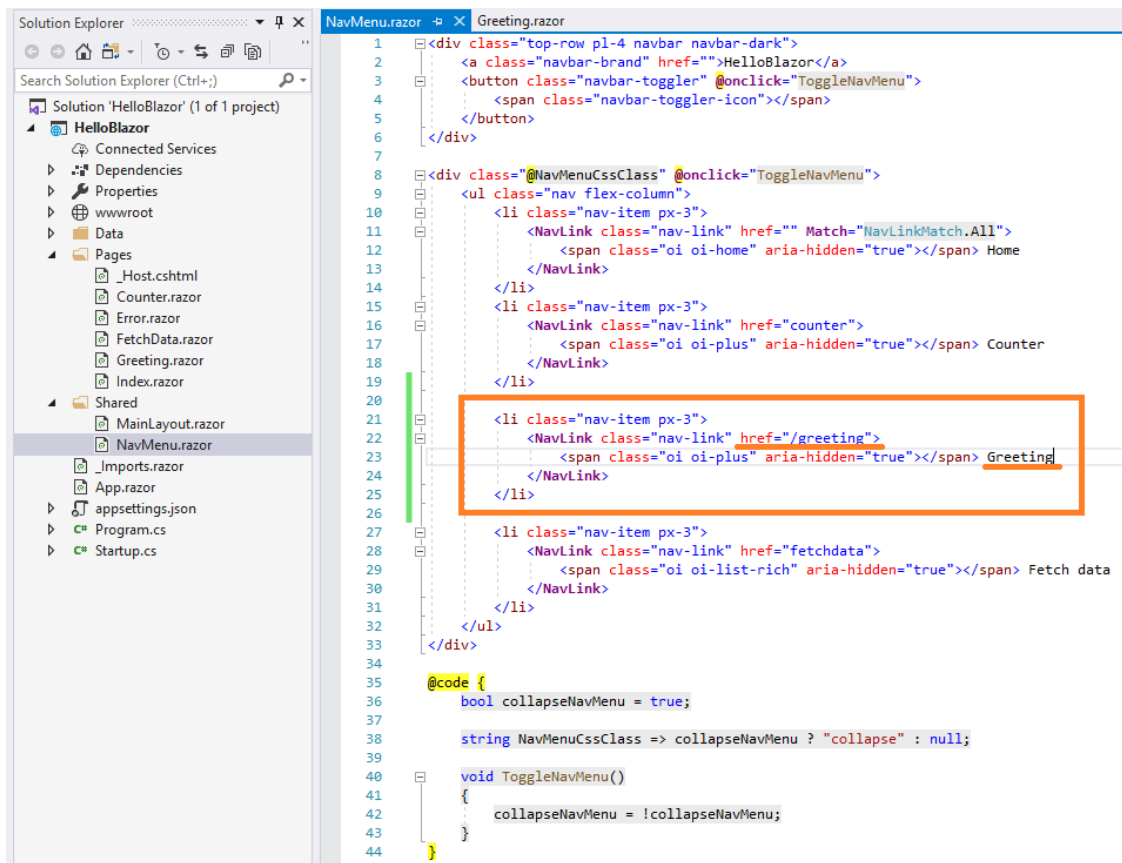
```

1  @page "/greeting"
2
3  <h1>@title</h1>
4
5  @code {
6      private string title = "Greeting from Blazor";
7      private string name;
8      private string address;
9      private string greeting;
10
11      private void Welcome()
12      {
13          if(!string.IsNullOrEmpty(name) && !string.IsNullOrEmpty(address))
14          {
15              greeting = $"Hello {name} from {address}. Welcome to your first Blazor app!";
16          }
17      }
18  }
19
20  <div class="form-group">
21      <input type="text" placeholder="Your name" @bind="name" class="form-control"/>
22  </div>
23  <div class="form-group">
24      <input type="text" placeholder="Your address" @bind="address" class="form-control" />
25  </div>
26  <div class="form-group">
27      <button @onclick="Welcome" class="btn btn-primary">Welcome!</button>
28  </div>
29  <h3>@greeting</h3>

```

\* Để bạn có cảm nhận riêng khi viết code, chúng tôi chỉ chụp ảnh code lên đây chứ không copy code text.

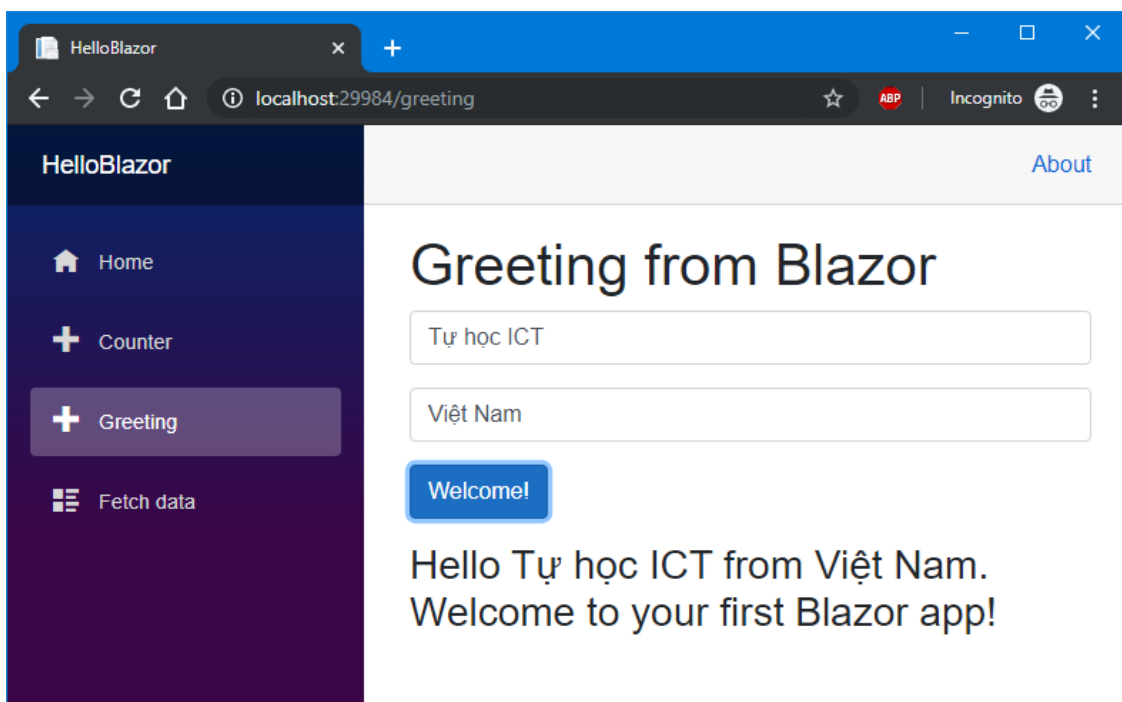
Sau đó mở file **NavMenu.razor** trong thư mục **Shared** và tạo thêm mục mới trong menu như sau:



```
1 <div class="top-row pl-4 navbar navbar-dark">
2   <a class="navbar-brand" href="">HelloBlazor</a>
3   <button class="navbar-toggler" @onclick="ToggleNavMenu">
4     <span class="navbar-toggler-icon"></span>
5   </button>
6 </div>
7
8 <div class="@NavMenuCssClass" @onclick="ToggleNavMenu">
9   <ul class="nav flex-column">
10    <li class="nav-item px-3">
11      <NavLink class="nav-link" href="" Match="NavLinkMatch.All">
12        <span class="oi oi-home" aria-hidden="true"></span> Home
13      </NavLink>
14    </li>
15    <li class="nav-item px-3">
16      <NavLink class="nav-link" href="counter">
17        <span class="oi oi-plus" aria-hidden="true"></span> Counter
18      </NavLink>
19    </li>
20    <li class="nav-item px-3">
21      <NavLink class="nav-link" href="/greeting">
22        <span class="oi oi-plus" aria-hidden="true"></span> Greeting
23      </NavLink>
24    </li>
25    <li class="nav-item px-3">
26      <NavLink class="nav-link" href="fetchdata">
27        <span class="oi oi-list-rich" aria-hidden="true"></span> Fetch data
28      </NavLink>
29    </li>
30  </ul>
31 </div>
32
33 @code {
34   bool collapseNavMenu = true;
35
36   string NavMenuCssClass => collapseNavMenu ? "collapse" : null;
37
38   void ToggleNavMenu()
39   {
40     collapseNavMenu = !collapseNavMenu;
41   }
42 }
```

Bạn chỉ đơn giản sao chép một mục có sẵn và điều chỉnh giá trị href – url tương đối tới trang razor vừa tạo và label của nó trên menu. Giá trị của href chính là chuỗi ký tự bạn đặt trong @page “/greeting”.

Chạy chương trình và bạn thu được kết quả như sau:












Nhập tên và địa chỉ vào hai textbox rồi ấn nút Welcome!. Bạn sẽ nhận được lời chào mừng từ Blazor!

Chúc mừng bạn đã xây dựng page component đầu tiên của Blazor!

Chúng ta có một số nhận xét ban đầu sau:

- Những gì trước đây bạn phải dùng JavaScript, giờ bạn có thể thực hiện hoàn toàn bằng C#!
- Các sự kiện được xử lý bằng phương thức của C#. Việc đọc và hiển thị dữ liệu động từ biến được thực hiện rất tiện lợi qua cơ chế databinding.
- Các kỹ thuật làm việc với Html và Css được giữ nguyên vẹn.
- Code C# và HTML markup trộn với nhau thoải mái nhờ cú pháp quen thuộc của Razor (nếu bạn từng làm việc với Asp.net MVC hoặc Asp.net Core).

Nếu mở DevTool (F12) của Chrome, bạn sẽ thấy các file được tải như sau:

Name	Status	Type	Initiator	Size	Time
 localhost	200	document	Other	2.5 KB	48 ms
 bootstrap.min.css	304	stylesheet	(index)	259 B	32 ms
 site.css	304	stylesheet	(index)	259 B	24 ms
 blazor.server.js	304	script	(index)	273 B	49 ms
 open-iconic-bootstrap.min.css	304	stylesheet	(index)	259 B	30 ms
 negotiate	200	xhr	blazor.server.js:1	412 B	20 ms
 open-iconic.woff	304	font	(index)	272 B	26 ms
 _blazor?id=SnFTu2m2YyHqsWMn9ZJq-A	101	websocket	blazor.server.js:1	0 B	Pend...
 favicon.ico	200	x-icon	Other	31.6 KB	15 ms

Bạn có thể thấy ứng dụng này rất đơn giản và gọn nhẹ. Ứng dụng không có quá nhiều file hỗ trợ, và cũng không có JavaScript (ngoại trừ blazor.server.js chúng ta sẽ nói đến sau).

Nếu giữ cửa sổ DevTool mở và bạn chuyển qua lại giữa các mục của menu, bạn cũng để ý thấy rằng các page hoàn toàn không được tải lại! Nói cách khác, client viết bằng Blazor giờ hoạt động tương tự như một SPA (Single Page Application) rất nhẹ nhàng.

Cũng lưu ý rằng, ở trên chúng ta vừa phát triển một **Blazor Server** app. Cơ chế làm việc của Blazor Server app rất khác biệt so với client web app viết bằng JavaScript.

## Dự án Blazor WebAssembly

Blazor còn có một dạng khác là **Blazor WebAssembly** (thường gọi tắt là Blazor Wasm). Blazor wasm phát hành cùng với Asp.net Core 3.1 vào tháng 5/2020.

Để làm việc với Blazor WebAssembly, bạn cần đến Visual Studio phiên bản tối thiểu là 16.6. Hãy cập nhật Visual Studio lên phiên bản mới nhất trước khi bắt đầu.

Hãy thực hiện lại các bước để tạo project Blazor giống như ở phần trên đã hướng dẫn. Ở bước "Create a new Blazor app", bạn lựa chọn **ASP.NET Core 3.1** thì template cho Blazor WebAssembly App sẽ xuất hiện.



Nếu chọn ASP.NET Core 3.0 thì chỉ có Blazor Server App, giống như ở phần trên đã thực hiện.

## Create a new Blazor app

ASP.NET Core 3.1



### Blazor Server App

A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This template can be used for web apps with rich dynamic user interfaces (UIs).



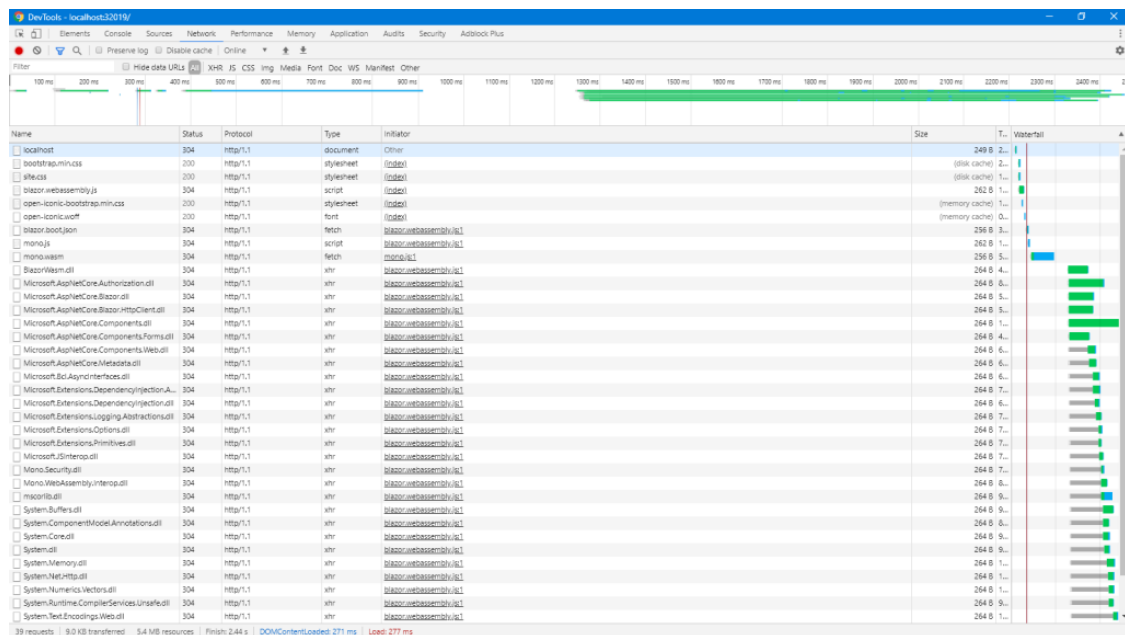
### Blazor WebAssembly App

A project template for creating a Blazor app that runs on WebAssembly. This template can be used for web apps with rich dynamic user interfaces (UIs).

Kết quả bạn sẽ thu được một project với cấu trúc rất giống như của Blazor Server đã làm ở trên. Nếu dịch và chạy bạn sẽ thu được kết quả y hệt.

Bạn có thể áp dụng lại những gì đã làm để tạo ra một page component mới giống như đã làm với Blazor Server.

Tuy nhiên, khi tải app trên trình duyệt bạn sẽ để ý thấy, Blazor WebAssembly tải chậm hơn khá nhiều. Nếu mở DevTools của Chrome bạn sẽ thấy vô số file được tải về như thế này:



Nếu để ý kỹ nữa thì bạn sẽ thấy, phần lớn các file tải về có đuôi dll (Dynamic Link Library – thư viện động). Một số file trong đó có tên rất quen thuộc như System.dll. Chúng ta sẽ giải thích vấn đề này trong một bài viết riêng để giúp bạn hiểu rõ hơn về mô hình hoạt động của Blazor webassembly.

+ Nếu bạn thấy site hữu ích, trước khi rời đi hãy **giúp đỡ** site bằng một hành động nhỏ để site có thể phát triển và phục vụ bạn tốt hơn.

+ Nếu bạn thấy bài viết hữu ích, hãy giúp **chia sẻ** tới mọi người.  
+ Nếu có thắc mắc hoặc cần trao đổi thêm, mời bạn viết trong phần **thảo luận** cuối trang.  
Cảm ơn bạn!

## Kết luận

Trong bài học này chúng ta bắt đầu làm quen với Blazor – framework mới của Microsoft cho phép viết ứng dụng đơn trang SPA trong Asp.net Core sử dụng C#.

Blazor hiện nay có hai mô hình phát triển: Blazor Server và Blazor WebAssembly.

Chúng ta sẽ phân tích chi tiết về sự khác biệt giữa hai mô hình ứng dụng này trong bài học tiếp theo.

Nếu muốn tìm hiểu thêm về kỹ thuật phát triển ứng dụng với Blazor, bạn có thể đọc ở trang tài liệu chính thức của Microsoft: <https://docs.microsoft.com/en-us/aspnet/core/blazor?view=aspnetcore-3.0>



**Tải mã nguồn Blazor**

1 file(s) 18Mb

TẢI MÃ NGUỒN