Como criar uma calculadora de IMC com ASP.NET Core Blazor

Rodolfo Ghiggi

- O Blazor é um framework, ou seja, uma ferramenta desenvolvida pela Microsoft que permite a criação de aplicações web modernas com interface rica, utilizando a linguagem C# tanto no back-end como no front-end.
  Imagine que uma empresa deseja desenvolver uma aplicação SPA (single page application, em uma tradução livre, aplicações de página única), e a equipe de pessoas que desenvolvem possui conhecimentos mais sólidos da linguagem C#. Nessa situação o Blazor pode ser uma ótima alternativa, pois com ele é possível desenvolver uma aplicação completa escrevendo apenas código C#.

Criando o primeiro aplicativo em Blazor

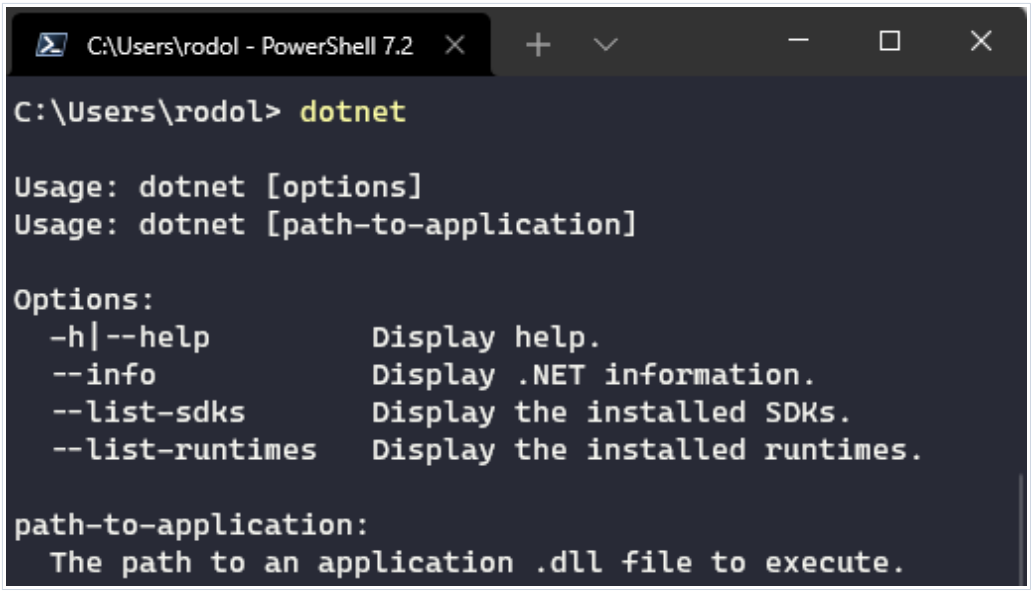
Vamos demonstrar como criar um aplicativo simples utilizando a versão BlazorServer.

Download e instalação

Para desenvolver aplicativos Blazor é necessário ter o .NET SDK instalado. Ele está disponível para Windows, Linux e macOS. Clique no link do seu sistema operacional para fazer a instalação.

•Instalador para Windows 64 bits
•Instalador para Windows 32 bits
•Instalar o .NET no Linux
•Instalador para macOS

Após a instalação, digite o comando dotnet no seu terminal, se a instalação tiver acontecido corretamente, um resultado parecido com esse deve ser exibido:



Criando a aplicação

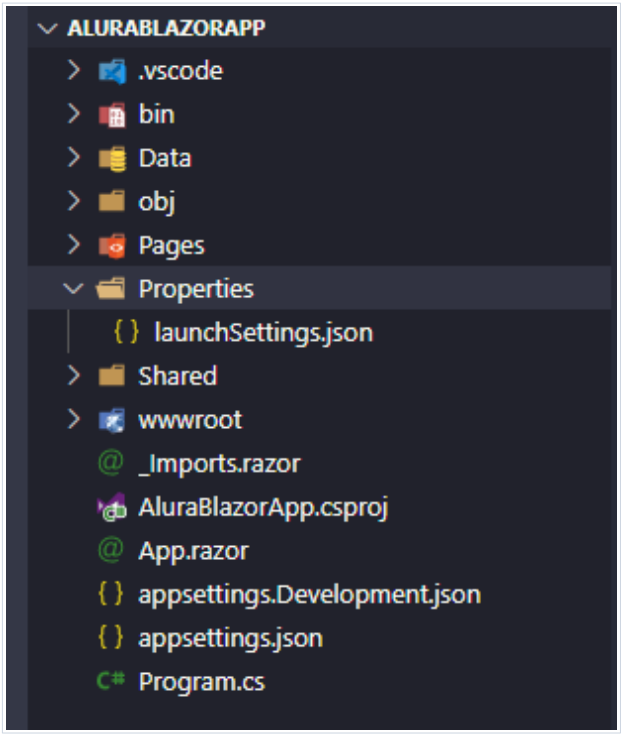
Para criar uma aplicação Blazor, digite o seguinte comando em seu terminal:

```
dotnet new blazorserver -o AluraBlazorApp --no-https
```

Dessa forma criaremos um projeto Blazor para a nossa aplicação em um novo diretório chamado “AluraBlazorApp”, dentro do diretório atual. Então, navegue para o diretório criado utilizando o comando:

```
cd AluraBlazorApp
```

Abaixo podemos encontrar uma rápida explicação sobre alguns dos principais arquivos do projeto Blazor.



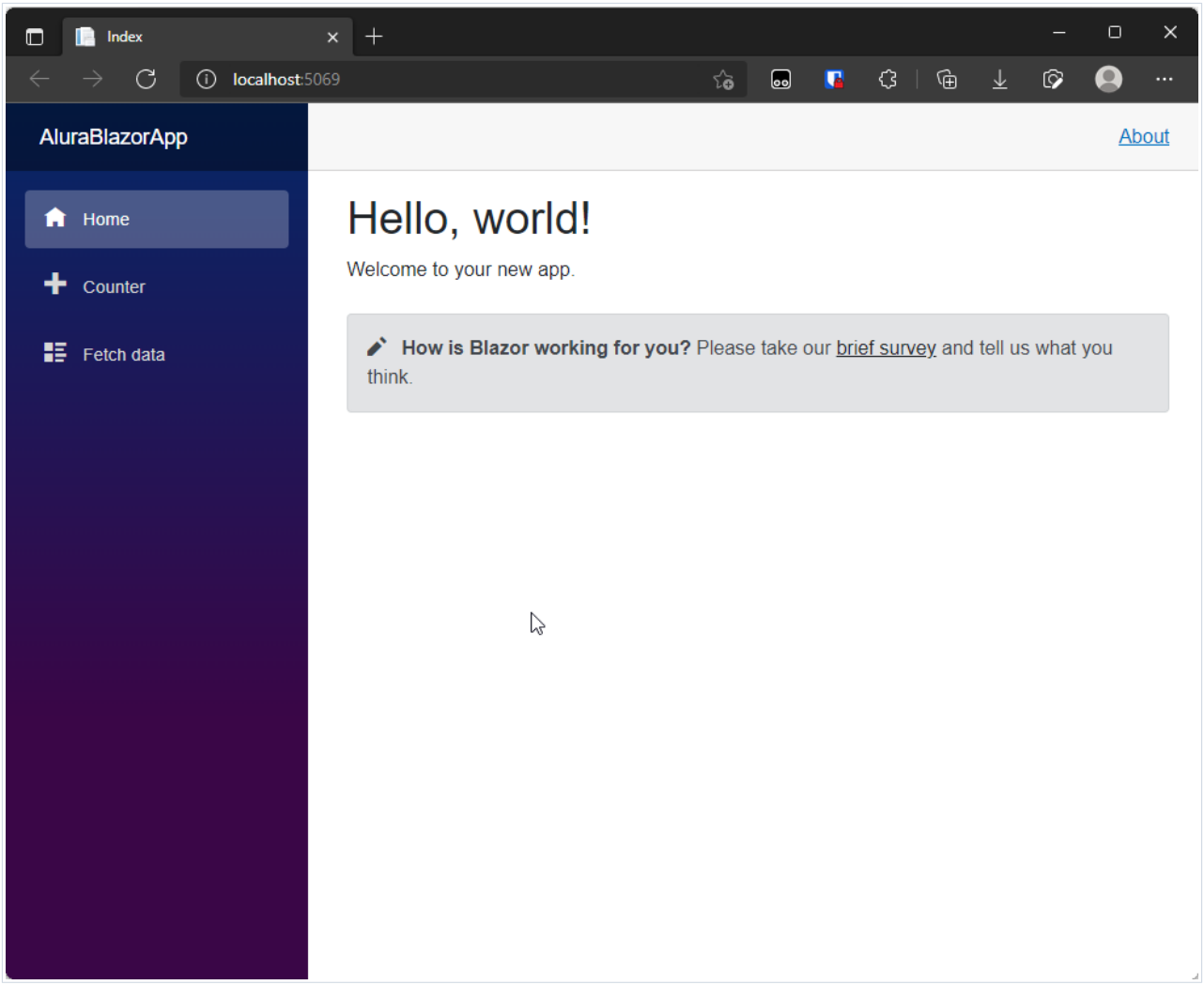
•`Program.cs` é o arquivo de entrada da aplicação, responsável por iniciar o servidor e configurar os serviços necessários.
•`App.razor` é o componente raíz da aplicação.
•`AluraBlazorApp.csproj` define o projeto e suas dependências.
•A pasta “Pages” contém alguns exemplos de páginas web.
O arquivo “launchSettings.json” , dentro da pasta “Properties”, define diferentes perfis para as variáveis de ambiente de desenvolvimento local. Durante a criação do projeto, um número de porta entre 5000 e 5300 foi automaticamente definido e salvo neste arquivo para ser utilizado em nosso projeto.

Executando a aplicação

Para executar a aplicação Blazor, digite no terminal:

```
dotnet watch
```

O comando `dotnet watch` constrói (ou seja, faz o build) e executa a aplicação. Assim, quando fizermos qualquer alteração no projeto, ele será automaticamente adotado na aplicação em execução. Caso queira interromper a execução, basta pressionar “Ctrl+C” no terminal.
Após alguns instantes, uma página como a da imagem abaixo deve ser exibida em seu navegador padrão.



O conteúdo exibido está definido no arquivo “Index.razor” na pasta “Pages”.

```
@page "/"

<PageTitle>Index</PageTitle>

<h1>Hello, world!</h1>

Welcome to your new app.

<SurveyPrompt Title="How is Blazor working for you?" />
```

Criando um componente

Agora chegou a parte mais divertida! Vamos criar um novo componente que será uma calculadora de IMC, o índice de massa corporal. O IMC é determinado pela

divisão da massa do indivíduo pelo quadrado de sua altura, com a massa em quilogramas e a altura em metros.

$$\mathrm{IMC} = \frac{\mathrm{massa}}{(\mathrm{altura} \cdot \mathrm{altura})}$$

Para criar um novo componente basta criarmos um arquivo .razor na pasta “Pages”. Então vamos criar o arquivo “Imc.razor”.

Na primeira linha do arquivo iremos definir a rota para o componente.

```
@page "/imc"
```

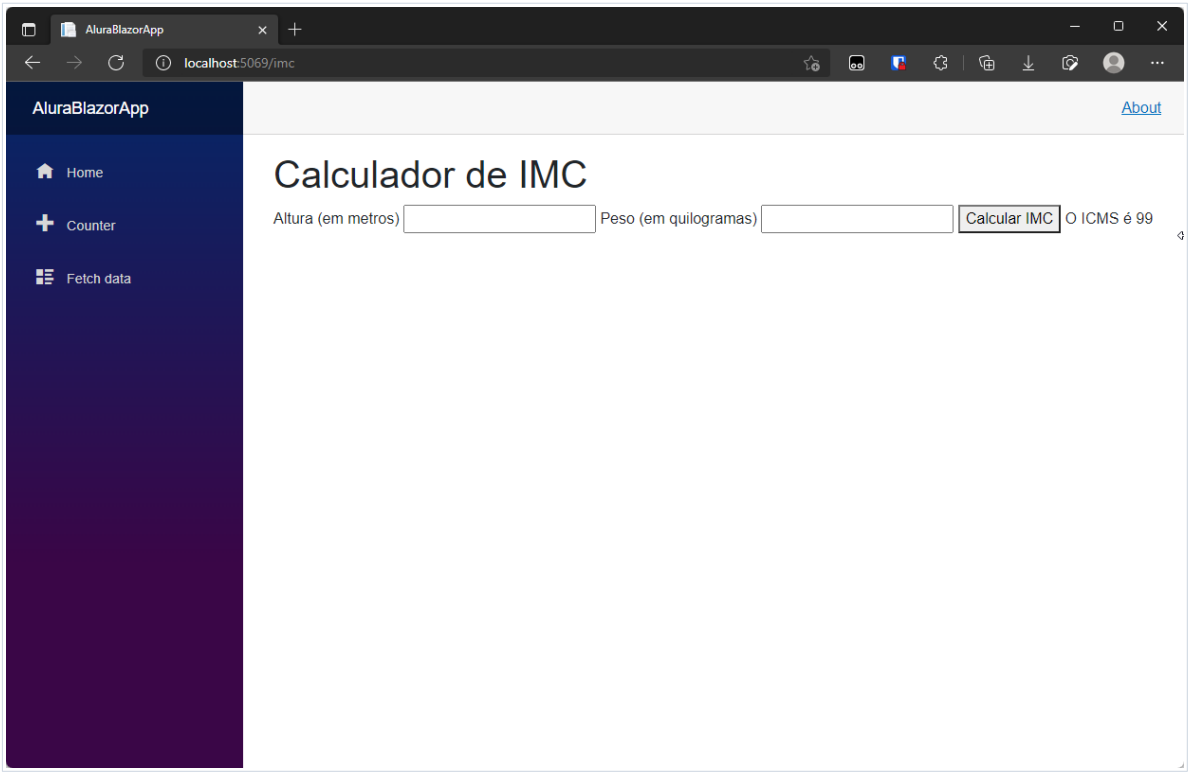
Em seguida, vamos incluir um título e dois inputs utilizando marcação HTML, um input para a altura e outro para o peso.

```
<h1>Calculador de IMC</h1>

<label for="altura">Altura (em metros)</label>
<input id="altura" type="number" />

<label for="peso">Peso (em quilogramas)</label>
<input id="peso" type="number" />
```

Vamos criar também um botão responsável por calcular o IMC e um label que irá exibir o resultado do cálculo.

```
<button>Calcular IMC</button>
<label>O IMC é <span>99</span></label>
```

Neste momento, se você acessar a rota/imc no seu navegador, deverá visualizar uma página parecida com a seguinte:



Implementando a lógica

Para escrever código C# em arquivos .razor, devemos declarar o bloco @code.

```
@code {

    // Nosso código C# ficará aqui

}
```

Primeiramente vamos declarar duas variáveis do tipo double para armazenar a altura e o peso.

```
@code {

    private double Altura { get; set; }

    private double Peso { get; set; }

}
```

Para vincular as variáveis aos inputs, usamos a diretiva @bind.

```
<label for="altura">Altura (em metros)</label>

<input @bind="Altura" id="altura" type="number"/>


<label for="peso">Peso (em quilogramas)</label>

<input @bind="Peso" id="peso" type="number" />
```

Agora que já temos as variáveis para altura e peso, vamos criar mais uma para armazenar o IMC calculado, além do método que será responsável pelo cálculo.

```
    private double ImcCalculado { get; set; }


    private void CalcularImc()
    {
        ImcCalculado = Peso / Math.Pow(Altura, 2);
    }
```

Feito o passo acima, precisamos chamar o método CalcularImc() quando o botão for clicado. Para isso devemos utilizar a diretiva @onclick="CalcularImc".

```
<button @onclick="CalcularImc">Calcular IMC</button>
```

Também precisamos alterar o span dentro do label de resultado para exibir o valor da variável ImcCalculado, em vez do valor fixo 99. Basta usarmos a diretiva @ImcCalculado.

```
<label>O IMC é <span>@ImcCalculado</span></label>
```

Se tudo ocorrer bem, seu aplicativo vai efetuar o cálculo com sucesso.

Segue o código completo do nosso componente:

```
@page "/imc"


<h1>Calculador de IMC</h1>


<label for="altura">Altura (em metros)</label>
<input @bind="Altura" id="altura" type="number" />
```

```
<label for="peso">Peso (em quilogramas)</label>
<input @bind="Peso" id="peso" type="number" />

<button @onclick="CalcularImc">Calcular IMC</button>
<label>O IMC é <span>@ImcCalculado</span></label>

@code {
    private double Altura { get; set; }
    private double Peso { get; set; }
    private double ImcCalculado { get; set; }

    private void CalcularImc()
    {
        ImcCalculado = Peso / Math.Pow(Altura, 2);
    }
}
```

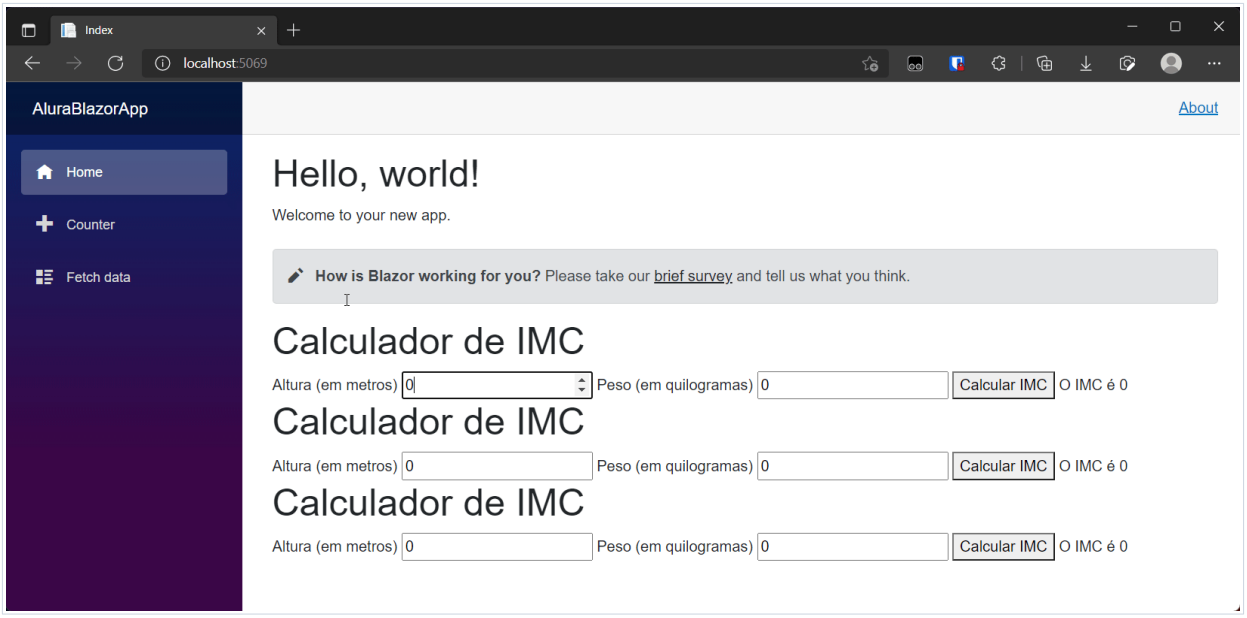
Reutilizando o componente

O nosso componente IMC pode ser reutilizado em outros componentes. Para adicioná-lo na página inicial, por exemplo, podemos digitar este código no arquivo Index.razor.

```
<Imc />
```

Podemos inclusive, adicionar várias vezes o mesmo componente e cada um funcionará de forma autônoma.

```
<Imc />
<Imc />
<Imc />
```



Este é o código completo do componente Index.razor.

```
@page "/"

<PageTitle>Index</PageTitle>

<h1>Hello, world!</h1>

Welcome to your new app.

<SurveyPrompt Title="How is Blazor working for you?" />

<Imc />
<Imc />
<Imc />
```

Conclusão

Nos últimos anos a Microsoft tem investido pesado no desenvolvimento do Blazor, um framework que já está sendo utilizado por diversos projetos em produção. Este artigo forneceu uma breve introdução ao framework Blazor, mostrando um exemplo de aplicação web para cálculo de IMC utilizando apenas a linguagem C#.

Caso queira se aprofundar mais no assunto

- Código fonte completo do projeto construído neste artigo
- Introdução ao ASP.NET Core Blazor

- ASP.NET Core
- Formação C# e orientação a objetos