

Microsoft.NET, HTTP get StringAsync, Async e Await

Transcrição

[00:00] Bom, agora que a nossa listagem está vazia, a gente vai ter que acessar o serviço para poder popular essa lista, para poder exibir na aplicação. Então, o primeiro passo aqui é pegar a URL do serviço, que a gente tem aqui na página, aqui no browser.

[00:18] Copiar essa URL e utilizar no nosso aplicativo, para poder acessar esses dados. Então, primeiro eu vou criar um método, que é o método novo, chamado get veículos, que vai servir para pegar a lista e adicionar na nossa listagem de veículos. Então, eu vou criar um método aqui público.

[00:43] Vou chamar de GetVeiculos() e aí, eu vou acessar aquela URL. Então, eu vou declarar aqui uma string, vou chamar de URL_GET_VEICULOS = "URL dos serviços", que é o aluracar.herokuapp.com. Bom, isso aqui, na verdade, seria melhor se fosse uma constante, então eu vou colocar como uma constante.

[01:16] E para melhorar, eu vou deixar isso lá em cima, lá no começo da classe, vou declarar aqui no corpo da classe, e aí, eu vou poder acessar a URL_GET_VEICULOS dentro do método GetVeiculos, que vai obter os veículos a partir do serviço.

[01:34] Agora, como que a gente faz para obter os veículos a partir do serviço? Bom, infelizmente o Xamarin Forms não tem uma maneira direta de acessar um serviço Http, porém, felizmente existem pacotes que permitem que a gente acesse um serviço Http.

[01:56] E a Microsoft tem um pacote desses que é o mais simples de utilizar, na minha opinião e a gente vai fazer uma instalação aqui, através do NuGet. Então, a gente procura aqui por Http, a primeira opção que vem aqui é o Microsoft.Net.Http.

[02:15] Então, eu vou selecionar somente aqui o primeiro projeto, que é o projeto test drive, que é o projeto portable, projeto portátil. Selecionando esse projeto, eu vou clicar em instalar, install, ele vai começar a instalar. Ele vai pedir para eu aceitar a licença. Então, eu aceito a licença e ele começa a instalação.

[02:42] Muito bem, já terminou de instalar. Então, já está pronto para a gente usar. Então, aqui dentro de GetVeiculos, eu vou poder acessar o client do http, que é o cliente, que é a biblioteca que permite que eu acesse funções, requisições do http.

[03:04] Então, eu vou instanciar aqui o client e ele é do tipo HttpClient, vou colocar new HttpClient e esse client, vou chamar... melhor, eu vou chamar ele de cliente. Então, esse cliente vai expor diversas funções, diversos métodos que permitem que eu acesse, que eu pegue informações ou que eu poste informações via protocolo http.

[03:37] Então, o método que a gente vai utilizar desses daqui, é o método que vai poder acessar a URL e fazer um get, como a gente viu antes, essa operação de entrar nesse endereço aluracar.herokuapp.com, um F5 aqui... Como a gente viu antes, isso aqui é uma operação get do http.

[04:04] Então, se a gente entrar, por exemplo, em Network e dar F5 de novo. Então, a gente clica aqui na chamada, eu venho aqui no header. Então, olha só, aqui eu tenho um request method, que é um método de requisição, chamado get. Então, a gente vai ter que utilizar um get para poder pegar...

[04:29] Para poder buscar as informações. Então, a gente volta lá no nosso código. Então, eu chamo... eu dou `GetStringAsync()` e vou passar aqui dentro qual é a URL que eu quero acessar. Então, a URL é aquela nossa constante, `URL_GET_VEICULOS`.

[04:49] Agora, olha só, se a gente olhar o nome desse método, `GetStringAsync`, o que a gente vê? Ele vai fazer um `get` e ele vai obter uma string. Essa string vai conter o resultado. Então, logicamente, ele vai retornar uma string para a gente. Só que olha só, a gente tem aqui `GetStringAsync`.

[05:10] Então, significa que esse resultado é assíncrono, ou seja, se a gente observar melhor aqui o método, a assinatura do método, se eu entrar aqui na definição com F12, eu vou chegar na assinatura do método, que eu estou vendo aqui que ele retorna, na verdade, não uma string, mas sim um `Task`.

[05:33] Então, se ele retorna um `task`, significa que o resultado pode demorar um pouco para chegar, ele vai executar uma tarefa e assim que estiver pronto o resultado, ele vai mandar de volta. Ou seja, se eu pegar o resultado desse `GetStringAsync` e simplesmente vir aqui em baixo...

[05:53] E colocar um código para utilizar, para popular a nossa lista, eu vou ter um resultado vazio, por quê? Porque ele não vai esperar o serviço executar e retornar os dados, para eu poder manipular esses dados, poder manipular essa lista. Então, para isso, eu preciso colocar aqui um modificador, uma palavra chave...

[06:18] Para que ele espere essa tarefa, essa `task`, para que ele espere que a tarefa seja concluída. Então, eu tenho que colocar um `await`, ele vai fazer com que esse método seja executado e ele só passe para a próxima linha, aqui em baixo do nosso método...

[06:44] Ele só vai continuar a execução depois que a tarefa lá da biblioteca `HttpClient`, depois que essa biblioteca retornar o resultado, depois que a tarefa for concluída. Então, assim, a gente vai poder continuar com o nosso código. Então, eu vou jogar aqui o resultado numa variável, que eu vou chamar de resultado mesmo.

[07:08] Então, eu vou colocar `var resultado = await client e o GetStringAsync`, que vai ser executado. Agora, quando eu faço isso, o Visual Studio está reclamando aqui, porque... o que acontece? Ele é um `awaitable`. Então, se eu estou colocando um `await`, significa que eu tenho que modificar a assinatura do meu método `GetVeiculos`.

[07:38] Eu tenho que modificar, para eu poder utilizar um `async`. Então, eu tenho que falar que esse método `GetVeiculos`, ele é um assíncrono. Então, eu vou ter que colocar `async` aqui e eu coloco um `task` também. Então, com `async Task`, eu estou marcando o `GetVeiculos` como assíncrono.

[08:03] E aí, eu vou começar a utilizar o resultado aqui em baixo. Então, vamos ver. Agora, eu quero fazer essa chamada, do método `GetVeiculos`, a partir lá do nosso `code behind`, da nossa página. Então, eu vou entrar em `listagem view`, lá no `code behind` e vou fazer a chama no `GetVeiculos`.

[08:27] E vou fazer isso aqui no evento `OnAppearing`. Então, eu vou acessar o `viewmodels` dessa `view`, a partir daqui. Mas como que eu posso acessar o `viewmodel`, se eu não tenho uma referência para ele? Então, o que a gente tinha feito antes, é declarar o `viewmodel` diretamente aqui no `xaml`.

[08:51] Então, vamos entrar aqui no `xmal`, olha só. Então, aqui dentro tem o `BindingContext` e aqui uma declaração para uma instância nova do `listagem viewmodel`. O que eu vou fazer é remover essa criação do `listagem viewmodel`, essa amarração com o contexto dessa página.

[09:17] E vou mover essa parte, eu vou mover lá para o `code behind`. Só que primeiro, eu preciso criar aqui uma propriedade, que eu vou chamar de `viewmodel`, que vai conter uma instância do `listagem viewmodel`. Então, `ListagemViewModel` e com essa propriedade, então...

[09:49] Eu vou primeiro instanciar, então, eu vou... Não, aqui não, eu vou ter que fazer isso lá no nosso construtor, então, eu faço `this.ViewModel = new ListagemViewModel`. Agora, no construtor, eu vou (setar) aqui para essa página qual é o contexto de binding.

[10:11] A gente estava fazendo isso no xaml, agora a gente está passando isso para o code behind, eu faço `this.BindingContext = this.ViewModel`, que é a instância do `ListagemViewModel`, que eu acabei de criar aqui em cima. Agora, no evento `OnAppearing`...

[10:29] O que eu vou fazer é utilizar essa instância do `ListagemViewModel`, para acessar o método que a gente criou lá no `viewmodel`, que é o `GetVeiculos`. Eu faço `this.ViewModel.GetVeiculos()`, aqui não tem parâmetro. O Visual Studio está me alertando aqui, que esse método `GetVeiculos`...

[11:00] Como ele é um assíncrono, posso querer colocar aqui um `await`, se eu quiser, eu posso colocar um `await`, para ele esperar a execução. Nesse caso, não vai ter muita diferença porque o nosso método aqui, ele termina nessa linha, então não ia fazer muita diferença.

[11:19] Ele está reclamando agora, porque eu estou usando o `await`, então eu tenho que colocar... eu tenho que marcar também esse método como um assíncrono. Então, eu vou deixar aqui uma palavra chave `async`, que vai indicar que esse método `OnAppearing`, agora, ele é assíncrono. Ficando: `protected async override void OnAppearing()`.

[11:37] E ele vai fazer essa chamada para o `GetVeiculos`, para popular a nossa lista de veículos. Então, o nosso método `GetVeiculos` ainda não está pronto, então a gente está começando a obter os dados que estão lá no serviço. Só que por enquanto, a gente já pode acessar alguma coisa.

[12:00] A gente já pode pegar o resultado, apesar de que esse resultado ainda não é o resultado final, o resultado que vai ser definitivo para a gente obter a lista. Só que a gente vai rodar a aplicação agora, só para ver o que a gente está obtendo com essa chamada para o `GetStringAsync`, que a gente está chamando aqui.

[12:20] Então, vamos ver o que o serviço vai retornar para a gente. Então, vai rodar o emulador agora. Está rodando a aplicação. Então, assim que a página da aplicação apareceu, ele chamou o método `GetVeiculos`. Então, vamos ver, vou colocar um breakpoint na segunda linha aqui embaixo, vou dar o F5.

[12:53] Olha só, ele executou a chamada lá no servidor, para obter a URL, que traz para a gente o Json. Então, a gente vai ver o que é esse resultado que ele trouxe aqui. Então, olha só, o resultado é uma string, como a gente esperava e essa string tem o formato Json.

[13:14] Então, olha só, a gente tem aqui o início com essa chave, a gente tem o início de uma listagem, de uma coleção, de um array. E aí, a gente tem uma chave indicante que tem um objeto e aí, começam as propriedades do objeto, nome, que é Azera V6, o preço, que é oitenta e cinco mil.

[13:33] Aí, vem aqui uma vírgula indicando que vai vir um outro objeto e aí, uma outra chave, o início do outro objeto. Aí, tem o nome, o preço, o nome e o preço. Então, a gente tem nessa string a representação de todos os objetos, de todos os veículos que estão vindo do nosso serviço.

[13:56] O que a gente vai fazer agora é começar a acessar, começar a manipular essa string, para transformar isso em objetos do sistema.