01

Apresentação

Transcrição

[00:00] E aí, pessoal, tudo bem com vocês? Eu sou o Marcelo Oliveira. Bem-vindos ao nosso curso do Xamarin com Visual Studio – parte 02. Na primeira parte do curso, a gente viu como construir uma aplicação de agendamento de teste drive para a empresa Alura Car.

[00:15] A gente fez isso utilizando o Xamarin Forms. Só que nessa parte 02, a gente vai prosseguir com o desenvolvimento dessa aplicação, melhorando alguns aspectos de arquitetura e a gente vai fazer isso atacando dois problemas básicos que são: o alto acoplamento e o isolamento da aplicação.

[00:35] Por que acoplamento? Bom, o auto acoplamento vai contra alguns princípios de designer de aplicações. Com o auto acoplamento, com muito acoplamento, a gente tem componentes que dependem muito uns dos outros. Então, para evitar isso, para facilitar o baixo acoplamento da aplicação...

[00:58] A gente vai utilizar um padrão de projeto chamado: MVVM ou Model View ViewModel. Então, hoje a gente já tem a view, que é a que apresenta os dados e a gente já tem o model, que é onde estão os dados do negócio e a lógica de apresentação.

[01:17] A gente já tem a lógica de negócio e os dados de negócio, a gente vai introduzir um componente intermediário chamado viewmodel, que vai fazer esse médio de campo. Então ele vai desacoplar a view do nosso modelo. A gente vai fazer isso usando o padrão MVVM.

[01:34] Em seguida, a gente vai ver como aumentar esse desacoplamento, permitindo que os componentes se comuniquem entre si, não diretamente, mas com um intermediário, que é o componente MessagingCenter do Xamarin Forms. Então, o MessagingCenter vai receber uma mensagem que foi enviada.

[01:55] E vai passar para frente, vai passar para outro componente. Então, com isso, os componentes da aplicação ficam menos dependentes entre si e continuando com o desacoplamento, a gente também vai remover a assinatura dos eventos de botão, o botão “clic”, por exemplo.

[02:16] A gente vai remover esses eventos e vai transformar eles em actions, em comandos que são componentes que vão estar na nossa classe viewmodel, que é a classe que vai ser criada com esse objetivo de desacoplar a aplicação. Em prosseguindo, a gente vai atacar o segundo problema, que é o isolamento da aplicação.

[02:39] Então, a aplicação do jeito que ela está, a gente trabalha com uma lista fixa nessa aplicação. Então, imagine que você tenha que incluir no veículo ou que você tenha que remover um veículo que não está mais disponível para o agendamento do test drive.

[02:54] Então, você teria que criar uma nova aplicação, criar uma nova versão, colocar nas lojas para os usuários baixarem essa nova versão e aí sim, ter a lista atualizada. Então, a gente não quer isso, a gente quer que os usuários automaticamente tenham essa lista atualizada.

[03:11] Então, por isso mesmo, a gente tem que utilizar alguma estratégia de acesso a dados remotos. Então, a gente vai fazer isso, utilizando Http GET, para puxar. A partir do servidor da Alura Car, a gente vai utilizar uma tecnologia para baixar essa lista de veículos disponíveis para o agendamento.

[03:32] Da mesma maneira, quando terminar o agendamento, a gente não vai deixar isso gravado somente no dispositivo do usuário, a gente quer mandar isso lá para o servidor da Alura Car, para oficializar o agendamento do test drive.

[03:46] Então, para isso, a gente vai usar a tecnologia Http POST, para poder salvar no servidor da Alura Car, os dados do agendamento desse veículo pelo usuário. Tá bom, gente? Então, espero que vocês gostem dessa parte do curso, dessa parte 02.

[04:03] Vamos ter mais duas partes, a parte 03 e 04, mais adiante, tá bom? Então, para essa parte 02, espero que vocês façam os exercícios, que vocês continuem prosseguindo nos estudos, tá bom? Ainda tem chão pela frente, espero que vocês curtam bastante essas novidades, tá bom?

[04:20] Então, muito obrigado. Até a próxima.