

Command e conclusão

Transcrição

[00:00] Bom, agora, para terminar então essa parte de MVVM, mensagem, mensageria e comandos da nossa aplicação, a gente vai fazer a mesma coisa para a última tela, que é a tela de agendamento. A gente vai entrar aqui no xaml, vai procurar aqui o Button, que é o botão agendar.

[00:20] A gente vai remover o evento clicked e vai trocar ela por um mando, então a gente vai colocar aqui Command, como a gente já viu antes, a gente utiliza o binding para amarrar o comando a um objeto comando, que vai existir lá no nosso viewmodel.

[00:38] Então, eu coloco aqui, AgendarCommand, que vai ser o nome do comando que vai existir no viewmodel. Então, lá no viewmodel, eu venho aqui no agendamento viewmodel, lá no final e vou colocar aqui, public, um ICommand, que é o tipo do comando, que é a interface do comando.

[01:06] Vou resolver a referência aqui, vou colocar, Agendar Command e vou colocar o get e o set, ficando: public ICommand AgendarCommand {get; set; }. Como a gente viu antes, a gente também tem que inicializar esse comando, não é simplesmente jogar a propriedade aqui e esperar que ela funcione.

[01:23] A gente tem que vir aqui no construtor, deixa eu procurar aqui o construtor, achei o construtor aqui em baixo. Então, a gente tem que inicializar esse comando. Então, agendar comando. Então, vai ser o new command, então é o novo comando.

[01:46] E aqui dentro, eu vou passar o método anônimo, uma expressão lambda aqui e dentro, eu vou fazer alguma coisa, eu vou tomar uma ação, baseado no clique do usuário, no botão de agendar. Então, o que eu vou fazer aqui? Eu vou enviar uma nova mensagem para a nossa central de mensagem do Xamarin Forms.

[02:10] Então, como que eu faço isso? Eu acesso MessagingCenter e faço o envio da mensagem, eu faço o send, que é o enviar em inglês. Aqui, eu tenho que passar também o tipo da mensagem que vai ser enviada. Que mensagem que eu vou enviar?

[02:31] Bom, eu vou enviar a mensagem do tipo do dado, do modelo que está sendo utilizado nessa página, que é um agendamento. Então, eu vou colocar aqui agendamento. Então, quando o usuário clicar aqui, a aplicação vai enviar uma mensagem do tipo agendamento.

[02:51] E aqui eu passo quem é o emissor dessa mensagem, que a referência diz, dessa classe e aqui o tipo da mensagem, eu vou chamar de agendamento mesmo. Então, aqui ele vai enviar a mensagem. Opa, aqui eu errei o parâmetro, é this.Agendamento.

[03:19] Agora sim, agora está certo. Ficou: MessagingCenter.Send(this.Agendamento, "Agendamento");. Ainda falta depois de enviar a mensagem, ainda falta assinar essa mensagem, subscrever essa mensagem, eu vou fazer isso lá no agendamento view.

[03:33] Então, no agendamento view, eu vou ter que colocar isso e algum lugar. Então, você lembra bem que a gente utiliza o método OnAppearing. Então, vamos colocar: protected override. Então, quando a página aparecer na tela, então a gente vai assinar essa mensagem de agendamento.

[03:57] Então, eu vou assinar a mensagem utilizando o MessagingCenter.Subscribe. E qual é o tipo da mensagem que a gente está recebendo aqui? É agendamento. Agendamento é o tipo da mensagem que vai ser passada para esse método aqui.

[04:15] Então... Ai, eu vou passar como argumentos, quem está assinando, que é a referência this para esse code behind e aqui o nome da mensagem, que é agendamento mesmo, ficando: MessagingCenter.Subscribe (this, "Agendamento").

[04:31] E aqui, eu passo... numa expressão lambda, eu passo aqui uma função anônima, que vai passar a mensagem para dentro de um código, que vai ser executado. Agora, que código que é esse? Aqui em cima, a gente tinha programado o evento clicked do botão.

[04:52] Então, o que a gente vai fazer é simplesmente mover esse código aqui para baixo. Eu vou recortar aqui, remover o evento clicked e vou colar aqui embaixo o nosso código, que vai exibir um alerta para o usuário, informando quais são os dados da mensagem.

[05:17] E claro que também falta fazer o código para desassinar, para cancelar a assinatura, para cancelar a subscrição dessa mensagem de agendamento. Então, a gente vai programar aqui, protected void override OnDisappearing, a gente vai cancelar essa assinatura.

[05:42] Agora, por que a gente faz isso mesmo? É porque a gente tem medo que uma mensagem seja assinada várias vezes, conforme a gente navega pela aplicação e essa mensagem, sendo assinada várias vezes, ela pode ser executada e tratada duas, três quatro vezes.

[06:01] Então a gente... é sempre bom colocar esse código de assinar ou cancelar assinatura, esses locais onde a gente tem certeza que uma assinatura vai ser cancelada, que ela vai ser anulada sempre que a página não estiver mais em uso. Então, aqui eu vou colocar MessagingCenter.

[06:21] E para cancelar a assinatura, eu utilizo Unsubscribe, eu passo aqui o tipo da mensagem que é agendamento, eu passo aqui a referência para quem assinou a mensagem que é this, que é essa própria classe e passo aqui o nome da mensagem, que é agendamento. Ficando: MessagingCenter.Unsubscribe(this, "Agendamento").

[06:40] Bom, agora tudo em ordem, a gente vai rodar a aplicação e ver como ela se comporta. Agora, rodando a aplicação. Agora, eu vou selecionar Fiesta 2.0, agora, vou apertar em próximo, vou preencher aqui: fulano de tal, vou colocar um telefone, 2345678, Email, fulano@gmail.com, vou dar o ok e vou agendar.

[07:18] Quando eu clico no agendamento, a mensagem que estava lá no.viewmodel, ela foi enviada pelo.viewmodel e foi capturada pela view, através do sistema de mensagens, do sistema de mensageria do Xamarin Forms. E aí, a view, pegando essa mensagem, ela soube tratar a mensagem...

[07:39] Exibindo aqui corretamente para a gente os dados do veículo que foram selecionados. Então, agora a gente vai fazer uma pequena revisão dessa aula. Então, o que a gente viu nessa aula aqui? A gente conseguiu se aprofundar no padrão MVVM.

[07:58] A gente conseguiu fazer com que a nossa view enviasse eventos para o.viewmodel diretamente, através do binding, tanto dos comandos, quanto do evento selected item do list view. Então, a gente viu também como a.viewmodel e a view, conseguem trocar informações através do sistema de mensageria do Xamarin Forms...

[08:27] Utilizando o quê? Usando o MessagingCenter. Então, a gente viu que com isso, a gente diminui o acoplamento, diminui a dependência entre os componentes. Então, isso ajuda a gente a deixar a nossa aplicação mais coesa, mais independente entre essas partes.

[08:48] Então, espero que vocês tenham gostado dessa aula, muito obrigado. Até a próxima.

