09

Agendamento service

Transcrição

[00:00] Agora que assinamos essa mensagem AgendamentoSelecionado, também temos que cancelar essa assinatura dessa mensagem, fazemos isso através do “override OnDisappearing”, eu vou criar um método aqui “override” procuro “OnDisappearing” e aqui dentro eu coloco o cancelamento dessa assinatura dessa mensagem.

[00:22] Então eu faço MessagingCenter.Unsubscribe, passa o tipo do objeto da mensagem que é “agendamento”, passa aqui a instância que é “this” e passo o nome identifica a mensagem, que é “AgendamentoSelecionado”.

[00:40] Agora voltando na Action, em que a gente envia o agendamento lá para o servidor da Aluracar, eu preciso fazer o quê? Preciso pegar o agendamento e fazer uma requisição “http post”, só que já fizemos isso uma vez, fizemos isso no começo do curso, quando enviamos na primeira vez que salvamos o agendamento, estamos enviando através de uma requisição http post.

[01:07] Só que isso está em outra classe, isso está no ViewModel lá do AgendamentoViewModel, se abrirmos aqui AgendamentoViewModel vamos conseguir encontrar, vamos conseguir localizar aqui embaixo o “SalvarAgendamento” que vai fazer esse trabalho, vai “serializar” os dados em formato “JSON” e vai criar uma instância “http client” e ele vai conseguir enviar o JSON, com os dados do agendamento lá para o servidor da Aluracar.

[01:42] Temos que colocar tudo isso em lugar centralizado para poder compartilhar esse código, eu não posso simplesmente copiar isso aqui, lá para o outro ViewModel e usar dois códigos, uma duplicação de código fazendo a mesma coisa, o que eu quero fazer aqui?

[01:59] Eu quero extrair esse método, não só o método “SalvarAgendamento”, mas também o método “SalvarAgendamentoDB” que vai fazer a gravação no banco de dados local do SQLite, eu quero extrair isso para uma outra classe, para mover esses dois métodos para uma outra classe, primeiro, vou fazer o quê?

[02:20] Eu preciso selecionar conteúdo do “SalvarAgendamento” e extrair para um outro método, porque o “SalvarAgendamento” vai ter que continuar aqui na classe “AgendamentoViewModel” só que eu tenho que mover só o conteúdo dele, porque só o conteúdo que vai ser extraído para uma outra classe.

[02:37] Vou fazer “Quick Actions & Refactorings”, para refatorar, agora eu clico em “Extrair método” e vou renomear aqui para “EnviarAgendamento”, agora eu tenho um novo método “EnviarAgendamento” e o método que já existia, que é o “SalvarAgendamentoDB”, eu vou criar uma nova classe aqui embaixo, que vou chamar de serviço de agendamento ou “AgendamentoService” e eu vou jogar esses dois métodos lá dentro.

[03:17] Só que o método EnviarAgendamento, como ele é consumido de fora da classe eu preciso mudar para o público. Eu vou ter que modificar esse método para eu passar para ele um parâmetro que é o agendamento, eu tenho que ter uma instância de agendamento sendo passada como parâmetro, para ele poder fazer aqui o envio lá para o servidor da Aluracar.

[03:42] Vou criar o parâmetro, “Agendamento”, e aqui é onde o Visual Studio está reclamando que ele não está encontrando a referência para várias propriedades do agendamento, eu preciso referenciar com essa instância de agendamento, “agendamento.DataAgendamento”, “DataAgendamento” aqui também, todos esses locais eu preciso substituir pela instância do agendamento, resolvendo aqui, agora essas propriedades também tem que vir do objeto agendamento.

[04:21] Aqui embaixo também, substituindo, aqui também, faltou tirar um ponto, aqui também e aqui também agendamento, quase tudo certo, agora faltou essa constante “UrlPostAgendamento” que ficou lá na ViewModel, eu vou remover da ViewModel e vou trazer aqui para nossa classe de serviço de agendamento.

[04:47] Movi a constante também, agora o método “SalvarAgendamentoDB” está reclamando que ele não está encontrando mais essa instância agendamento aqui, vamos passar ela também por parâmetro, “Agendamento agendamento” aqui embaixo na hora de salvar e vou passar essa instância do agendamento e aqui no “SalvarAgendamentoDB” eu tenho que passar também a instância do agendamento.

[05:20] Agora voltando para ViewModel, esse método enviar agendamento está dando erro porque ele não existe mais aqui na ViewModel, ele faz parte de uma outra classe, o que eu preciso fazer?

[05:33] Eu preciso instanciar o “AgendamentoService agendamentoservice = new AgendamentoService” e eu vou chamar um método “EnviarAgendamento” que está na instância de “AgendamentoService”. Só que quando eu envio agendamento eu preciso passar uma instância de agendamento, então “this.Agendamento”, estou passando a instância local da ViewModel, lá para “AgendamentoService” para ele enviar os dados.

[06:07] Agora para ficar mais organizado eu vou mover essa classe “AgendamentoService” para um outro arquivo, vou criar aqui uma pasta de serviços, adicionar nova pasta, Services e eu vou criar uma nova classe aqui, nova classe que vou chamar de “AgendamentoService”, agora é só mover a classe lá para dentro, copiando aqui e colando em seguida.

[06:40] Agora eu preciso acertar aqui essas referências, estou importando os “using”, preciso arrumar aqui também, agendamento também, agendamentoDAO também. Faltou aqui o JsonConvert e agora podemos usar essa instância também lá no outro código. Aqui eu preciso acertar também. Agora eu preciso acertar também a chamada que estamos fazendo para o serviço de agendamento para poder reenviar os dados para o servidor da Aluracar.

[07:20] Agora nesse código eu vou instanciar o “AgendamentoService agendamentoservice = new AgendamentoService” e eu vou chamar um método que vai enviar o agendamento, só que dessa vez estamos reenviando os dados, estamos pegando aqui o quê? O agendamento que está vindo na mensagem de agendamento selecionado, que é aquele que falhou da primeira vez, vamos reenviar ele.

[07:59] Estou enviando aqui, esse “EnviarAgendamento” é uma task, então eu vou colocar aqui um “await” para esperar o resultado dele, agora vamos rodar a aplicação e ver se conseguimos reenviar esses dados lá para o servidor da Aluracar, quando eu clico nesse botão eu vou para tela, deu um erro aqui, provavelmente é porque tem algum problema com a nossa nova propriedade que é “AgendamentoSelecionado”.

[08:28] Eu vou abrir aqui a ViewModel, venho aqui “AgendamentoSelecionado” é isso mesmo, eu fiz uma besteira aqui, antes de enviar essa mensagem lá para o “MessagingCenter” eu preciso primeiro “settar” o “AgendamentoSelecionado”, agora está na ordem certa, só que antes disso eu também tenho que verificar se o valor que está vindo aqui no “set” seria não nulo, eu só posso passar aqui para esse código se o valor for diferente de nulo.

[09:02] Agora sim, só vamos enviar a mensagem do agendamento que foi selecionado se ele não for nulo. Vamos rodar a aplicação de novo, vou entrar com o usuário “joao@alura.com.br”, senha “alura123” e vou entrar, agora eu vou lá para tela de meus agendamentos, agora está certinho, agora vou criar mais alguns agendamentos até eles começarem a falhar.

[09:29] Legal, temos dois aqui para reenviar, esses dois que estão em vermelho, que é o Azera e o C3, vamos clicar aqui. Tem um problema aqui, reenviar, deseja reenviar o agendamento, não. Quando eu clico nesses que estão em preto, quer dizer, que deram sucesso no primeiro envio, ele está perguntando se eu quero reenviar, o que eu preciso fazer aqui é verificar se o agendamento ainda não foi confirmado, se ele já foi confirmado eu não posso reenviar, porque ele já está salvo no servidor da Aluracar.

[10:06] Vou ver aqui no código e onde eu estou perguntando se deseja reenviar eu preciso verificar primeiro se o agendamento não foi confirmado, se o agendamento for confirmado ele nem entra aqui, eu tenho que verificar essa negativa, o agendamento não pode ser confirmado para eu poder reenviar. Vamos rodar de novo. Agora eu vou clicar nesses que já foram enviados com sucesso, não aparece nenhuma mensagem.

[10:37] Agora vou clicar nesses que estão em vermelho que deu falha no agendamento, agora vou confirmar o reenvio dos dados, sim, só que não teve nenhuma mudança na lista, porque a lista não alterou?

[10:50] Porque eu não pedi para alterar, estamos fazendo a mudança, enviando os dados para o servidor, gravando no banco SQLite, mas em que momento eu estou atualizando a minha lista da ViewModel? Não estou pedindo em lugar nenhum, eu vou fazer isso agora.

[11:07] Como eu estou aqui no Code Behind da página "AgendamentoUsuárioView", eu preciso acessar a ViewModel e pedir para a ViewModel atualizar a lista para mim, eu faço "this.ViewModel", ViewModel não é uma propriedade, não tem uma instância ViewModel nessa classe, nessa Code Behind, o que preciso fazer é criar uma ViewModel local, estamos referenciando o BindingContext diretamente a uma nova instância da ViewModel dele.

[11:43] Eu vou criar aqui um novo tipo, que eu vou colocar como ReadOnly e aqui View Model, eu coloco ReadOnly porque ela só vai ser instanciada uma vez e vou chamar essa instância, essa propriedade local, esse atributo local de ViewModel, primeiro vou instanciar "this.*viewModel = new" e aqui uma nova instância da ViewModel, aqui no BindingContex eu vou "settar" esse "this.*viewModel", agora eu posso utilizar o "this._viewModel" para poder chamar um método para atualizar a lista para mim.

[12:30] Assim que o agendamento é enviado para o servidor com sucesso, ou com falha, vamos atualizar a lista que está na View Model. O que eu tenho que chamar?

[12:44] Eu tenho que chamar a lista, só que eu tenho que chamar um método que atualiza a lista, esse método não existe ainda, eu preciso criar um método novo, quando entramos no construtor da ViewModel temos o código que vai buscar os dados no SQLite e montar a lista para nós, em vez de colocar todo esse código dentro do construtor, até para melhorar o aspecto dele, para deixar ele mais limpo, vamos extrair um método.

[13:16] Vamos lá, "Quick Actions & Refactorings", vamos extrair um método aqui, eu vou chamar de "AtualizarLista", agora eu posso chamar esse método AtualizarLista lá do meu Code Behind, só que primeiro eu preciso deixar ele público, "public". Agora eu já posso chamar ele de fora, voltando lá para o Code Behind eu vou fazer "this.viewModel.AtualizarLista", AtualizarLista vai ser chamado para podermos ver a lista atualizada, com os dados novos do envio de agendamento.

[13:54] Vamos rodar a aplicação de novo, agora eu vou colocar aqui, vou escolher meus agendamentos, vou pegar o agendamento que falhou, Azera V6, deseja reenviar agendamento, sim, vamos dar uma olhada, apareceu aqui embaixo, Azera V6, vamos reenviar esse C3, deseja reenviar? Sim, mas será que atualizou essa lista? Vamos ver aqui, apareceu o C3 aqui embaixo, reenviado, mas já temos um C3 aqui em cima, se eu clicar de novo, desejo reenviar, sim, apareceu mais uma linha com C3.

[14:34] Mas não é isso que eu quero, eu preciso que essa linha que está em vermelho mude para preto, indicando que houve sucesso no reenvio, eu não quero que apareça uma nova linha de agendamento, vamos ter que modificar esse código para poder atualizar exatamente a linha que estamos reenviando.