08

Outros SwitchCell e revisão

Transcrição

Para que a ativação ou desativação de todos os acessórios afetem o valor total (ação que ainda não está ocorrendo), precisaremos modificar o *code behind*, implementando-se as outras propriedades, atualizando-as conforme necessário.

Inicialmente, incluímos o código referente ao freio ABS, agora faremos o mesmo para o ar-condicionado e MP3 player. Em `DetalheView.xaml.cs`, portanto, acrescentaremos:

```
bool temArCondicionado
public bool TemArCondicionado
{
    get
    {
        return temArCondicionado;
    }
    set
    {
        temArCondicionado = value;
        OnPropertyChanged();
        OnPropertyChanged(nameof(ValorTotal));
    }
}

bool temMP3Player
public bool TemMP3Player
{
    get
    {
        return temMP3Player;
    }
    set
    {
        temMP3Player = value;
        OnPropertyChanged();
        OnPropertyChanged(nameof(ValorTotal));
    }
}
```

Agora precisamos alterar o `ValorTotal` de acordo com as outras *flags*, os valores boolean que indicam se o ar-condicionado ou MP3 player estão ativados. No mesmo arquivo:

```
return string.Format("Valor Total: R$ {0}",
Veiculo.Preco
+ (TemFreioABS ? FREIO_ABS : 0)
+ (TemArCondicionado ? AR_CONDICIONADO : 0)
+ (TemMP3Player ? MP3_PLAYER : 0)
);
```

Feito isto, rodaremos a aplicação verificando o que ocorre. Quando ativamos as chaves dos outros acessórios, porém, o valor total não é alterado. Ao acessarmos `DetalheView.xaml`, veremos que não colocamos o `Binding` para a propriedade `On` em todos os acessórios, aquilo que indica se o `SwitchCell` está ligado ou não. Faremos isto agora:

```
<TableSection Title="Acessórios">
    <SwitchCell Text="{Binding TextoFreioABS}" On="{Binding TemFreioABS, Mode=TwoWay}"></SwitchC
    <SwitchCell Text="{Binding TextoArCondicionado, Mode=TwoWay}" On="{Binding TemArCondicionadc
    <SwitchCell Text="{Binding TextoMP3Player, Mode=TwoWay}" On="{Binding TemMP3Player, Mode=Twc
    <TextCell Text="{Binding ValorTotal, Mode=TwoWay}"></TextCell>
</TableSection>
```

Rodaremos novamente a aplicação para verificar o que acontece agora. Selecionamos "Fiesta 2.0" e também o freio ABS, desativando-o em seguida. Faremos o mesmo com o ar-condicionado, para testarmos e veremos que o preço se altera, como esperado. A mudança ocorre assim que ativamos a chave do MP3 player também.

Recapitulando o que foi visto: implementamos a *view* de detalhes do veículo selecionado pelo usuário, criamos o `TableView`, que colocamos em um `StackLayout`, junto ao botão "Próximo", na mesma tela.

Configuramos a margem do `StackLayout` por meio da propriedade `Padding`, muito comum para quem trabalha com webdesign, HTML e CSS.

Vimos como utilizar um `TableView`, cuja intenção definimos como sendo `Settings`, que é "configuração". Com qualquer *smartphone* em mãos, você verá as configurações em uma tela semelhante à que implementamos, com agrupamento, texto em cima (os `TableSections`) nas seções. Neste caso há apenas uma `TableSection`, mas poderiam ser várias.

Em `TableSections`, conseguimos configurar os acessórios do veículo utilizando o `Intent`, a intenção de `Settings`. Aprendemos que em um `TableView` não podemos colocar qualquer controle ou componente, como botão ou `Label`. É necessário utilizar tipos de componentes específicos do Xamarin Forms: as células, *Component Cells*.

Os `SwitchCell`s são os três componentes (acessórios), os quais fazem o papel do *checkbox*, existentes geralmente em aplicações desktop. Temos os `SwitchCells` em *smartphones*, com esta aparência de chaves. Vimos que é possível colocar textos em um `SwitchCell`, o que pode ser feito até em um `Binding`.

Utilizamos também o `Binding` na propriedade `On`, que indica se o *switch* está ativo ou não. Onde se vê o valor total, colocamos outra célula, `TextCell`, indicada para se exibir texto em um `TableView`.

Finalmente, o ponto mais importante visto foi como utilizar o `Binding` para fazer a atualização de algum outro ponto da interface, no caso, trata-se do `TextCell` que contém o valor total atualizado do veículo. Conforme vamos alterando o `SwitchCell` dos acessórios, também modificaremos o valor total. Isto servirá de gancho para seguirmos mais adiante.

Espero que tenham gostado, muito obrigado, até a próxima!