02

TableView e TableSections

Transcrição

Começaremos a implementar a terceira tela do aplicativo, referente ao agendamento e cadastro do usuário. Em `AgendamentoView.xaml`, que é o código XAML, teremos a tela em formato de formulário. Vimos anteriormente que para isto utilizaremos o `TableView`.

Criaremos um `TableView` com `TableRoot`, sua raiz, cujas seções agruparão os dados do formulário. Ao observarmos os desenhos dos layouts que queremos implementar, temos a seção "Seus dados" e, abaixo, "Agendamento". Estes grupos são duas seções distintas contidas em um `TableView`, ou seja, trata-se das `TableSections`, as quais teremos que nomear utilizando-se `Title`.

```
<TableView>
    <TableRoot>
        <TableSection Title="Seus Dados">
        </TableSection>
        <TableSection Title="Agendamento">
        </TableSection>
    </TableRoot>
</TableView>
```

Vamos rodar a aplicação para ver como isto aparece no emulador do Android, passando por todas as telas até chegarmos à terceira. Os `TableViews` que acabamos de implementar são exibidos corretamente. Incluiremos mais detalhes em cada uma das `TableSections`, sendo que no primeiro precisaremos colocar campos de entrada em relação a nome, telefone e e-mail do usuário, indicados por `Label`s.

```
<TableView>
    <TableRoot>
        <TableSection Title="Seus Dados">
            <EntryCell Label="Nome:"></EntryCell>
        </TableSection>
        <TableSection Title="Agendamento">
        </TableSection>
    </TableRoot>
</TableView>
```

Rodando a aplicação, vê-se que o primeiro `EntryCell` aparece com o `Label` "Nome:" exibindo-se o campo de entrada conforme gostaríamos. Preencheremos este primeiro campo com "fulano de tal", apenas por motivos de teste. Vamos fazer o mesmo em relação ao campo "Fone:", rodando a app em seguida.

```
<TableView>
    <TableRoot>
        <TableSection Title="Seus Dados">
            <EntryCell Label="Nome:"></EntryCell>
            <EntryCell Label="Fone:"></EntryCell>
        </TableSection>
        <TableSection Title="Agendamento">
        </TableSection>
```

```
    </TableRoot>
</TableView>
```

Um teclado **alfanumérico** (genérico, com letras e números) é exibido quando vamos digitar o telefone, clicaremos na tecla correspondente a eles, o que não é ideal ao usuário, pois isto não é conveniente e acaba sendo um trabalho a mais. Para resolver este problema, vamos criar uma opção em `EntryCell` que definirá o teclado adequado para este tipo de campo.

Acrescentaremos, portanto, uma propriedade chamada `Keyboard`, que quer dizer "teclado" em inglês. Quando rodamos a app, esperamos que este teclado mude para um formato mais amigável para o usuário.

```
<TableView>
    <TableRoot>
        <TableSection Title="Seus Dados">
            <EntryCell Label="Nome:"></EntryCell>
            <EntryCell Label="Fone:" Keyboard="Telephone"></EntryCell>
        </TableSection>
        <TableSection Title="Agendamento">
        </TableSection>
    </TableRoot>
</TableView>
```

Na última tela digitaremos "fulano de tal" no campo destinado ao nome e, ao clicarmos no campo a ser preenchido com o telefone, o teclado ativo contém apenas números. Vamos incluir o próximo campo de entrada de dados, para o e-mail:

```
<TableView>
    <TableRoot>
        <TableSection Title="Seus Dados">
            <EntryCell Label="Nome:"></EntryCell>
            <EntryCell Label="Fone:" Keyboard="Telephone"></EntryCell>
            <EntryCell Label="E-mail:"></EntryCell>
        </TableSection>
        <TableSection Title="Agendamento">
        </TableSection>
    </TableRoot>
</TableView>
```

Feito isto, vamos rodar a aplicação novamente. Quando vamos digitar o e-mail, mais uma vez aparece o teclado genérico, por ser o modo *default* do teclado. No entanto, todo e-mail deve possuir um símbolo de arroba (`@`), acessível apenas quando clicamos na tecla `?123`.

Gostaríamos que este símbolo aparecesse no primeiro teclado. Assim como fizemos com o telefone, alteraremos o teclado em relação ao e-mail pela propriedade `Keyboard`, para deixá-lo mais adequado, facilitando seu uso:

```
<EntryCell Label="E-mail:" Keyboard="Email"></EntryCell>
```