

AGENDA 13

Imagens e
Componentes
Diversos de Interface
Gráfica



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autores:

TIAGO ANTONIO DA SILVA

KELLY CRISTIANE DE OLIVEIRA DAL POZZO

São Paulo – SP, 2024

Ao desenvolver aplicativos com .NET MAUI, o uso de imagens e componentes de interface gráfica é fundamental para criar uma experiência de usuário intuitiva e visualmente atraente. Com .NET MAUI, é possível integrar facilmente imagens em diversos formatos (como PNG, JPEG e SVG) diretamente no layout, utilizando componentes como Image e ImageButton. Além disso, os componentes visuais disponíveis, como Button, Entry, Label, Grid, e StackLayout, permitem construir interfaces dinâmicas e responsivas.

Um conceito importante para conectar esses componentes visuais com os dados e a lógica do aplicativo é o **BindingContext**. Ele é usado para associar elementos da interface com propriedades ou modelos de dados, permitindo que as informações sejam automaticamente refletidas na interface do usuário. Por exemplo, ao definir o BindingContext de uma página ou componente para um modelo de dados, é possível fazer com que as mudanças nesse modelo atualizem a interface de maneira automática e sem a necessidade de código adicional.

Essa técnica de data binding simplifica o desenvolvimento ao separar a lógica de apresentação dos dados, proporcionando uma manutenção mais fácil e facilitando a adoção de padrões arquiteturais como o MVVM (Model-View-ViewModel).

A combinação de imagens, componentes gráficos e a vinculação eficiente de dados via BindingContext permite a criação de aplicativos robustos, modernos e otimizados para uma ampla gama de dispositivos, garantindo uma navegação fluida e uma experiência rica para o usuário final.

Assista o vídeo oficial da Microsoft que fala sobre o assunto:

<https://learn.microsoft.com/en-us/shows/dotnet-maui-for-beginners/dotnet-maui-data-binding-with-mvvm-xaml-5-of-8-dotnet-maui-for-beginners>



Nesta agenda iniciaremos o desenvolvimento do projeto App Hotel. O objetivo é explorar recursos de interface gráfica e aprofundar os conhecimentos nos recursos da linguagem C# e funcionalidades da XAML.

Vamos explorar a relação entre os elementos de interface gráfica e o código C# e para isso estudaremos um dos conceitos mais úteis e importantes presentes na .NET MAUI: Binding Context. Essa ferramenta auxilia a comunicação entre o front-end do App e as regras de negócio implementadas no back-end.

Binding Context

No .NET MAUI, a vinculação de elementos XAML via BindingContext é uma técnica fundamental para conectar a interface do usuário aos dados ou à lógica de negócios, utilizando data binding. O BindingContext atua como o contexto que fornece os dados necessários para os elementos visuais de uma página ou controle. Isso permite que as informações exibidas na interface sejam atualizadas automaticamente conforme os dados mudam, facilitando a separação entre a lógica do aplicativo e a apresentação visual. Essa abordagem melhora a organização do código e promove uma interface de usuário mais dinâmica e reativa. Quando você define o BindingContext de uma página ou controle no XAML ou no código-behind, todos os elementos descendentes dentro daquela página ou controle podem acessar e vincular-se a dados contidos no contexto. Esse conceito permite que os elementos da UI (interface gráfica) sejam atualizados automaticamente quando os dados no contexto mudam, como iremos fazer

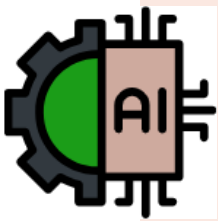
nessa agenda, enviando dados de uma tela à outra usando o BindingContext.

Como funciona a definição?

Ao definir o BindingContext, você especifica um objeto de origem de dados para a página ou controle. Cada elemento filho pode então usar a sintaxe de vinculação {Binding NomeDaPropriedade} para se conectar a propriedades do objeto definido no BindingContext.

A vinculação pode ser configurada tanto em XAML quanto no código-behind. O .NET MAUI suporta vários modos de vinculação:

- ✓ **OneWay:** O valor da fonte de dados é atualizado na UI, mas mudanças na UI não afetam a fonte de dados.
- ✓ **TwoWay:** Atualizações na UI também atualizam a fonte de dados e vice-versa. Esse é o comportamento usado no exemplo com o Entry.
- ✓ **OneTime:** O valor é vinculado apenas uma vez, no momento da criação da UI.



Utilize ferramentas de Inteligência Artificial para pesquisar mais sobre o conceito de BindingContext, pesquise sobre as principais vantagens de aplicação do conceito. A IA pode ajudar a encontrar rapidamente informações atualizadas e exemplos práticos.

Componente Stepper

O elemento Stepper no .NET MAUI é um controle que permite ao usuário selecionar um valor numérico dentro de um intervalo predefinido. Ele é composto por dois botões: um para aumentar e outro para diminuir o valor. O Stepper é útil em cenários onde se deseja permitir ajustes incrementais de um valor, como a escolha de uma quantidade de produtos, controle de volume, ou ajuste de valores numéricos em geral.

Algumas das principais propriedades do Stepper:

- ✓ **Minimum:** Define o valor mínimo que o Stepper pode atingir.
- ✓ **Maximum:** Define o valor máximo que o Stepper pode atingir.
- ✓ **Increment:** Define o valor pelo qual o Stepper aumenta ou diminui a cada clique.
- ✓ **Value:** O valor atual do Stepper. Essa propriedade pode ser vinculada a um ViewModel, facilitando o data binding.
- ✓ **ValueChanged:** Um evento que é disparado sempre que o valor do Stepper muda.

Veja um exemplo prático de utilização:

```
<Stepper Minimum="0" Maximum="10" Increment="1" Value="5"
    ValueChanged="OnStepperValueChanged"/>
```

Alguns exemplos para aplicação do componente Stepper:

- ✓ Definir quantidades em um formulário de pedido.
- ✓ Ajustar parâmetros de configuração (ex: nível de volume, brilho, etc.).
- ✓ Navegação entre páginas ou controle de opções de um aplicativo.

O componente DatePicker é um controle que permite aos usuários selecionar uma data de forma interativa por meio de uma interface visual. Ele exibe a data selecionada como texto e, ao ser clicado, abre uma interface onde o usuário pode navegar por um calendário e escolher a data desejada. Esse componente é muito útil em formulários e em situações onde o usuário precisa escolher uma data específica, como datas de nascimento, agendamentos ou prazos.

As principais propriedades do DatePicker são:

- ✓ **Date:** A data atualmente selecionada no controle. O valor padrão é a data atual.
- ✓ **MinimumDate:** Define a menor data que o usuário pode selecionar.
- ✓ **MaximumDate:** Define a maior data que o usuário pode selecionar.
- ✓ **Format:** Define o formato no qual a data será exibida como texto, por exemplo, "dd/MM/yyyy" ou "MM/dd/yyyy".
- ✓ **DateSelected:** Um evento que é acionado quando o usuário escolhe uma nova data.

A propriedade Format do DatePicker define como a data será exibida no controle. Você pode usar diferentes padrões de formatação, como:

- ✓ **"dd/MM/yyyy":** Exibe a data no formato "dia/mês/ano".
- ✓ **"MM/dd/yyyy":** Exibe a data no formato "mês/dia/ano".
- ✓ **"D":** Exibe a data por extenso, como "terça-feira, 10 de outubro de 2024" (depende da cultura configurada).

O Picker é um componente que permite ao usuário selecionar um item de uma lista suspensa. Sendo ideal para quando o usuário precisa fazer uma escolha entre várias opções pré-definidas, como em formulários para escolher uma categoria, uma cor, um país ou qualquer outra lista de valores.

O Picker é semelhante a um dropdown ou combo box em outras plataformas e pode ser vinculado a uma lista de opções ou a uma coleção de objetos. O Picker também pode ser facilmente integrado com a arquitetura MVVM, usando data binding para vincular as opções e o valor selecionado.

Algumas propriedades do componente Picker:

- ✓ **ItemsSource:** Define a lista de itens que serão exibidos no Picker. Pode ser uma lista de strings, uma coleção de objetos ou qualquer coleção.
- ✓ **SelectedIndex:** O índice do item atualmente selecionado. Se nenhum item estiver selecionado, o valor é -1.
- ✓ **SelectedItem:** O item atualmente selecionado. Pode ser vinculado a uma propriedade do ViewModel.
- ✓ **Title:** Define o texto que será exibido no Picker quando nenhum item estiver selecionado.
- ✓ **ItemDisplayBinding:** Permite que você especifique como os objetos complexos serão exibidos no Picker, definindo uma propriedade específica para exibir no controle.

Assista a vídeo aula explicativa onde foi aplicado em um projeto todos conceitos apresentados até o momento:



Disponível em: <https://www.youtube.com/watch?v=DvFpB63U8TI>

CÓDIGO-FONTE COMPLETO

<https://github.com/tiagotas/MauiAppHotel/tree/f4651e5172dedfceec4630538ffe4bf340bce128>